Simulating Stellar Hydrodynamics at Extreme Scale

Paul R. Woodward
University of Minnesota

Falk Herwig
University of Victoria

Ted Wetherbee
Fond du Lac Tribal and Community College

Simulating the hydrogen ingestion flash in asymptotic giant branch stars is discussed to illustrate a computational science research problem demanding high-performance computation at scale. The relation of this work to the National Strategic Computing Initiative's objectives is discussed.

Over the past several years, we have been simulating brief but important events in the evolution of stars that stretch our ability to harness the power of present large computing systems and that will continue to put demanding requirements on future systems. These events are brief, in that they last only a few days, or even less, while typical time scales for stars are very much longer. Nevertheless, they are not as brief as explosive events, and for this reason, we must simulate the behavior of the star for a great many dynamical times. These events also depend critically upon processes that are inherently 3D. The events we simulate occur in deep convection zones that develop above nuclear burning shells. We find that the most important modes of convection in these shells are comparable in size to the entire shells themselves, forcing us to include in our simulation domain the entire interior region of the star, not just a small section of it. Our simulations therefore present the challenge that we must describe the whole stellar interior over a long time interval, which requires millions of time steps. Because we are unwilling to wait the necessary time for these simulations to complete on a small cluster of machines—which would be years—we must find a way to get all this done in a reasonable time by using large numbers of tightly coupled machines in a large, single computing system. We must find ways to have enormous numbers of computational engines simultaneously involved productively in the work, despite the fact that the greatest challenge is the number of time steps needed rather than the number of spatial grid cells. We will briefly describe the phenomena we study, and then describe some of the strategies we have employed to both scale the code up to use nearly 14,000 nodes, each with 32 CPU cores, and to run on each of these CPU cores very efficiently.

Our work relates to the National Strategic Computing Initiative (NSCI) mainly through its first objective, which is to achieve capable exascale computing. In line with the second NSCI objective, we are also combining into our single application code data analysis and visualization functions and finding ways to have these run efficiently as integrated parts of the simulation on a single computing system. Finally, our work in developing simulation software that can take advantage of the latest high-performance computing (HPC) systems at extreme scale relates to objective 4 in the NSCI strategic plan, which seeks to develop a more powerful, flexible, and inclusive software and hardware ecosystem to more fully exploit the promise of computing in scientific discovery, economic competitiveness, and national security.

STELLAR HYDRODYNAMICS OF THE HYDROGEN INGESTION FLASH

We are interested in the generation of heavy elements in stars, especially as this bears upon the chemical evolution of galaxies in the early universe.[1–4] The development of planetary systems like our own requires the enrichment of the interstellar gas from which they are formed by gas from earlier generations of stars that have produced heavy elements. Understanding the generation of the elements in stars is therefore part of understanding the story of our origins. Quantitative predictions of the formation of these elements in stars are based largely on 1D numerical simulations of stellar evolution. Nuclear reactions near or at the centers of stars produce heavy elements, and for these to ultimately be expelled into the interstellar medium, either the star as a whole must explode or the heavy elements must be transported into the outer envelope of the star, where the gas can be blown off by stellar winds. The transport of material radially within a star is treated in such calculations by approximate, spherically averaged, 1D models of convection, despite it being an inherently 3-D process. Convection zones form where either very large amounts of heat due to nuclear energy production are deposited that radiative heat diffusion is unable to carry outward fast enough, or where the opacity of the material changes so that radiative heat diffusion's effectiveness is reduced. Inside convection zones, over the time scales in which they persist, the material becomes well mixed, so that the effect for material transport is easy to model. However, problems arise at the edges of convection zones, where the gas changes from a nearly adiabatic stratification (representing marginal convective stability/instability) to a stable stratification, in which entropy increases outwards. It is at these convection zone boundaries that we need a fully 3D description of the convection to find the extent to which stably stratified fluid is entrained into the convection flow. This entrainment of stably stratified gas, a process we call convective boundary mixing (CBM), can, under special conditions, have very large effects.

An example of the importance of CBM for stellar evolution and generation of heavy elements is the hydrogen ingestion flash (HIF).[5–6] This can occur in a late stage of evolution for an intermediate mass star. In the phase of their evolution after core helium burning, asymptotic giant branch (AGB) stars have periodic outbursts called thermal pulses. Each such thermal pulse begins with helium that is accumulating between the hydrogen-burning shell and the degenerate carbon-oxygen core suddenly igniting in what is called the helium shell flash.[7] The energy generation in the just-ignited helium burning shell rises so rapidly at this point that the energy cannot be carried outward effectively by radiative heat transport, and therefore a convection zone develops above the helium burning shell. This convection zone is called the pulse-driven convection zone (PDCZ). The PDCZ grows in mass by incorporating more and more of the gas above it, and it also grows in radius by lifting the overlying layers upward. This upward lifting of the hydrogen burning shell reduces its temperature by an expansion of the gas, and the hydrogen burning effectively ceases. What is special about this sequence of events in AGB stars of the early universe (as well as in some post-AGB stars of the present universe[8]) is that the PDCZ can grow by incorporating gas from above it right up to the point where it encounters gas where the hydrogen has not yet burned.

In a 1D calculation of this process, entrainment of this hydrogen-rich gas from above the convection zone boundary is produced by a simplified 1D model that represents this as a diffusion process.[9] After about a year of this entrainment and growing nuclear energy generation, the entropy barrier between the convection zone gas and the H-rich envelope above is breached. At this point, the 1D representation of the flow cannot be trusted. These changes in the star occur deep within its interior, close to its tiny and extremely dense carbon-oxygen core. A chain of reactions beginning with burning of ingested hydrogen results in the production of free neutrons, which are then captured on a whole series of heavy nuclei to produce a spectrum of heavy elements.

This process has specific element abundance signatures that can be identified in certain very metal-poor stars.

Simulating the Hydrogen Ingestion Flash in 3D

We here present results of a large-scale, 3D simulation of the HIF, starting with the 1D stellar structure model of a 2-$M_\odot$ AGB star with a very low metal content typical for stars of the early universe. This simulation serves as an example of the kind of investigation that is enabled by very large, tightly interconnected computing systems like the one we used at NCSA, the Blue Waters machine. It is important to stress that our simulations presented here compute the entrainment and transport of very small concentrations of hydrogen-rich gas that ultimately have very large effects, as we can see in Figure 1. The entrainment of the H-rich gas occurs at a boundary that separates gas that is unstable to convection from stably stratified gas above it. This boundary is very stiff. In the deep stellar interior, convective transport is very efficient, the stratification inside the convection zone is almost adiabatic, and accordingly the speed of the pulse driven convection flow is low, with typical Mach numbers ranging from 0.01 to 0.035. The slow flow together with the steep entropy gradient at the top of the convection zone, signaling the strong stability of the layer above the boundary, contribute to the stiffness of this convection zone boundary. Because the fluids are gases, of course, some level of entrainment is inevitable. The questions are: what level, and how does this occur? These questions cannot be answered by 1D simulations.

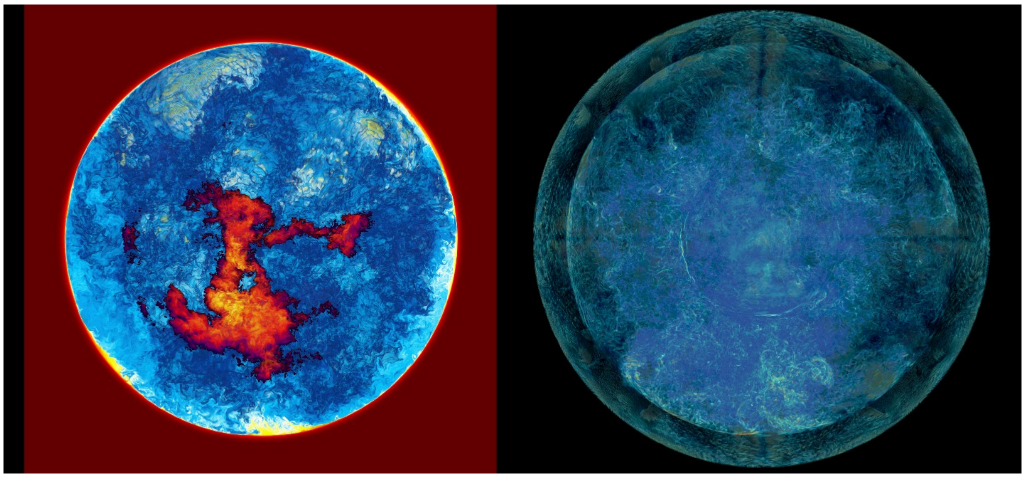

Figure 1. Two volume-rendered views of the far hemisphere of the simulated interior of a 2 $M_\odot$ AGB star of the early universe, with metallicity $Z = 10^{-5}$. Left: The concentration of H-rich gas entrained into the convection zone above the helium-burning shell during the H-ingestion flash. The material of the convection zone and of the degenerate carbon-oxygen (C-O) core is transparent. Where the clipping plane passing through the center of the star meets the top of the convection zone, a clearly visible circle in both images, we see the entrained gas in cross section. Outside the circle, the concentration is unity, and it appears red. As the concentrations decrease, colors go from red to yellow, white, aqua, and finally, dark blue. We see the dome of the spherical convection zone boundary in projection, and where descending entrained H is burning, we represent the released energy by colors that indicate a flame, with the energy increasing as the colors go from purple to red and finally to yellow and white. Right: The magnitude of vorticity. Vorticity is confined mainly to the inside of the convection zone, from about radius 11 to 28 Mm. The bottom of this convection zone, which is clearly visible as a circular feature, is very near the surface of the C-O core. Between the top of the convection zone and the outer bounding, reflecting sphere, there is very little gas motion, even though shear right at the bounding sphere makes it visible in this very sensitive variable. A burst of strong vorticity and high turbulence is visible near the bottom of the image, where a rising gust driven by recent burning of ingested H is pushing against the top of the convection zone and driving wedges of entrained gas visible in yellow at the bottom of the left image. This is the beginning of a GOSH (see main text).

Figures 1–4 show selected frames from a stereo movie (available at www.lcse.umn.edu) of the approach to and the process of the HIF in the simulation. The 3D simulation begins at about half a year before the HIF in the 1D simulation. We cannot advance the star in 3D for so long a time, so we chose to speed up the evolution by increasing the luminosity from helium burning that is driving the convection. Multiple series of lower-resolution 3D simulations with a variety of different luminosities show that the entrainment rate for gas from above the convection zone varies linearly with this luminosity to a high degree of accuracy.[10] This scaling relationship can be understood by the argument that it takes a certain amount of energy to pull the more buoyant entrained gas down against its tendency to rise, and that the more rapidly we supply energy, the more rapidly we will pull gas down.[11] We chose to increase the luminosity of the star by a factor of 22.5, which should make the HIF arrive in a few days rather than a few months. The convective energy flux scales with the third power of the velocity. Hence, our increase in the luminosity drives up the velocity magnitude by a factor of 2.8, and the resulting velocities are still of modest Mach numbers ranging between 0.03 and 0.10. Since the Mach numbers remain low, the character of the flow does not change much, and therefore, we believe that our simulation is a good approximation to a flow with the proper luminosity that has simply been sped up.

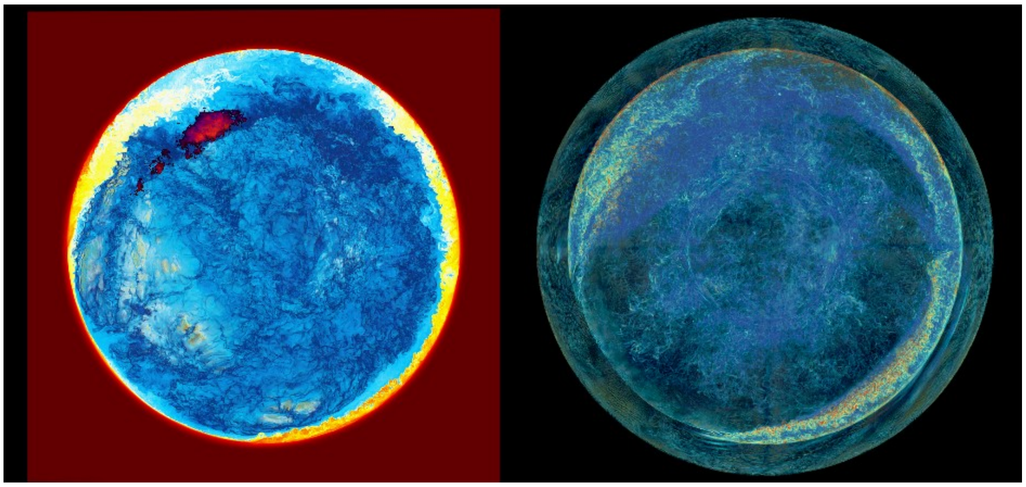
Figure 2. A global, tsunami-like wave driven by burning ingested hydrogen traverses the star.

In Figures 1–4, we have selected times to display that show how the HIF proceeds, and that make clear that this process is very different from a 1D picture that we might have expected from stellar evolution code studies, such as the picture laid out by Falk Herwig and his colleagues in 2011.[8] The rendering parameters and colors are explained in Figure 1's caption. At each time, we show the ingested H concentration on the left and the vorticity magnitude on the right. The left-hand image of each pair shows the distribution of the fuel for the HIF, with a representation of the H-flame superposed, and the right-hand image shows the velocity field that is bringing this fuel into the flame zone, which lies about halfway down in the convection zone. For the first 888 minutes of simulated time, we did not let the ingested hydrogen burn. This gave us a measure of the ingestion rate without the influence of positive feedback from the burning of the ingested gas. Once we let the ingested gas burn, we found that the ingestion rate increased by more than two orders of magnitude. The heat released from burning gas in a descending plume about halfway to the bottom of the convection zone is powerful enough to reverse this downward flow, as can clearly be seen in the movie from which the images in Figures 1–4 have been taken. In a 1D spherically symmetric picture, ingested gas would burn at the same level, but could not do so in a localized fashion, and it therefore could not drive the tsunami-like waves that act like plows to bring relatively huge amounts of newly ingested gas into the convection zone. Such waves can be identified in Figure 2 from the wedge-like formations of enhanced H-concentration visible where our clipping plane slices through the star, and we see the region near the top of the convection zone in cross section.

The result of the greatly enhanced H-ingestion from these propagating waves that travel all the way around the star, which we have termed the GOSH[12] (global oscillation of shell hydrogen ingestion), is the formation of a highly H-enriched layer that floats on top of the convection zone. This layer is visible in Figure 3. In a 1D spherically symmetric picture, this layer would become a new convection zone. The H would burn at its bottom, and reaction products of H-burning could not go down into the original convection zone, the PDCZ, below it. Our 3D simulation, shown in Figures 3 and 4, reveals that the burning of the H at the bottom of this floating H-rich layer is overstable. Rising convective plumes from below push the bottom of this upper layer upward, and the difficulty of this fluid to rise nearly this much above the level of the top of the original convection zone, where the entropy barrier is much greater, forces the layer downward at the opposite side of the star. The downward motion of the H-rich gas at that opposite location causes much more rapid burning at that location, because the reaction rate is so sensitive to the temperature, which increases with depth in the convection zone. This in turn creates an even stronger upwelling, setting an oscillation—a GOSH—in motion. This is beginning to happen in Figure 4, and the ultimate result is a complete disruption of the layer, with violent H-combustion. This final behavior, shown at its beginning in Figure 4, makes our confining and bounding reflecting sphere, visible as the outermost sphere in the vorticity images of Figures 1–4, strongly influence the simulation for the first time. For this reason, we do not trust the continuation of the simulation beyond this point, after which all the hydrogen above the original convection zone was rapidly consumed.

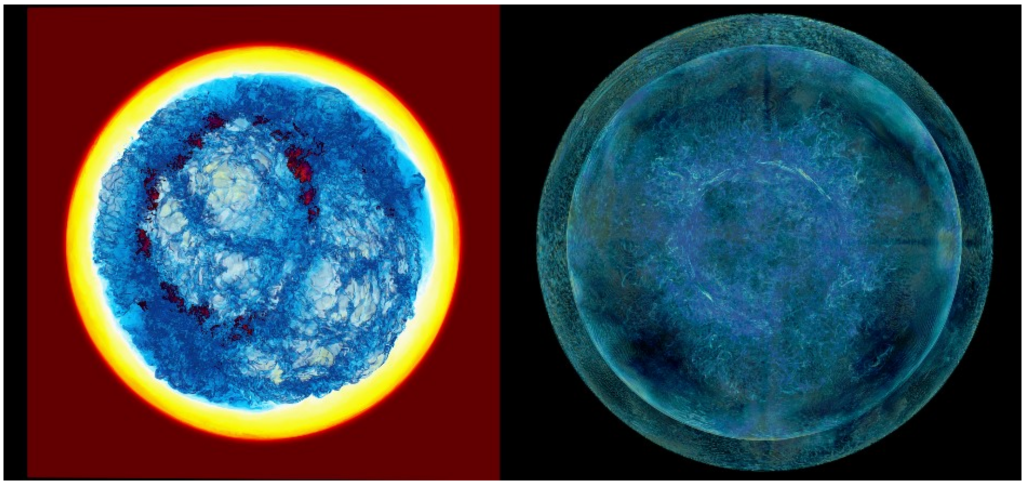
Figure 3. The GOSH (see main text) has created a hydrogen enriched layer floating at the top of the convection zone, driven by helium burning in the turbulent gas near the C-O core.

To get a better grasp of how the HIF proceeds and to assess its ultimate effects on the star and the nucleosynthesis of heavy elements, we must modify our simulation program substantially to describe the layers outside of the region studied so far and to move our bounding sphere significantly outward to allow the region of the PDCZ to expand. The challenges involved with designing a code like ours to perform simulations of this sort at the resolution used in Figures 1–4 and running the problem in a reasonable time are discussed in the next section. From the example of the HIF just described, it should be evident that full 3D simulations of this sort of phenomenon are needed to take the evolution of a star through events of this sort, which can be viewed quite literally as life-changing events.

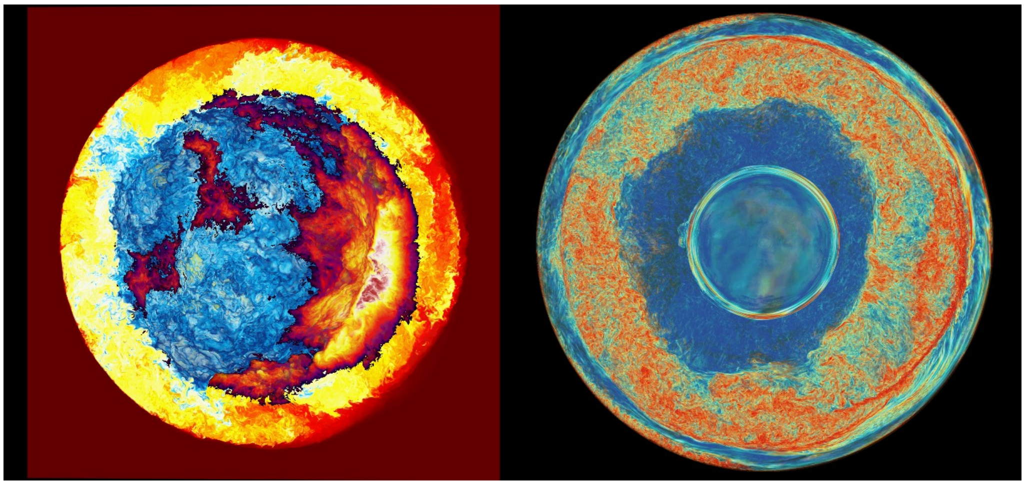

Figure 4. The same two volume-rendered views of the far hemisphere of the simulated interior of a 2 $M_\odot$ AGB star of the early universe, with metallicity $Z = 10^{-5}$, but now the renderings are at time 2,699 min. A violent wave of combustion of entrained, H-rich gas is now seen at the lower right, about to cross the star, pulling down large concentrations of H-rich fuel. At this stage, the presence of our outer bounding sphere becomes strongly felt, and our simulation can no longer be trusted. All the hydrogen within the bounding sphere is quickly consumed after the combustion wave reaches the opposite side of the star. We are now testing a new code that will be able to move the bounding sphere outward a factor of 2 in radius.

ILLUSTRATION OF THE SOFTWARE AND HARDWARE ECOSYSTEM'S IMPORTANCE

The simulation shown in Figures 1–4 was computed on a uniform Cartesian grid of $1{,}536^3$ cells. These 3.6 billion cells would have been a large grid in 2000, but certainly in 2018, this is a modest grid indeed. In an early simulation on NCSA's Blue Waters machine in late 2012, for example, we used a grid of $10{,}560^3$ cells on a very different scientific problem.[13] The grid used in Figures 1–4 is 325 times smaller, but because we must advance the solution in Figures 1–4 for 9 million time steps instead of the other problem's 85,000 time steps, the amount of computational labor is actually comparable. In an earlier time, we would have sought a special computing system with especially fast CPUs to do our stellar hydrodynamic problems. In 1985, for example, a Cray-built processor was roughly 150 times faster than the standard department-level machine, the DEC VAX. As Larry Smarr memorably noted at that time, the difference between computing on a VAX and on a Cray is like the difference between walking and flying in a jet plane. If you choose to walk to California from New York, you will eventually get there, but you would be extremely unlikely to go there at all unless you could fly. This is why certain kinds of scientific research are unlikely to happen at all unless computing systems that enable the work to be done in days rather than years are provided. In 1985, one could simply buy a Cray, and researchers could move their codes there and, almost effortlessly, begin to fly. Today, single processing cores that are 150 times faster than standard ones in an academic department simply do not exist. Researchers must now make special efforts in the design of their codes to use machines built from hundreds of thousands of CPU cores, as we discuss below. Without these machines, they would most likely choose smaller and less difficult problems to attack, with the advance of our scientific understanding suffering accordingly.

The Impact of Amdahl's Law

We have been doing the great bulk of our computing on the Blue Waters machine at NCSA. Blue Waters has about 26,000 nodes, with each node containing four CPUs, each with eight CPU cores. The key consideration in getting up to 14,000 such nodes to work productively on the same, not-so-large problem was identified decades ago: it is called Amdahl's law. Though quite a simple idea, it is very difficult to address in a large scientific computation. Amdahl pointed out in the late 1960s that if the computation contains some small fraction of work that must be performed serially and cannot be spread across many individual processors, then the rate of overall progress of the computation can be strictly limited by the time required to perform this serial labor. The larger the number of cooperating CPU cores we devote to a computation, the harder it becomes to keep this serial portion of the work from holding everything up and thus dominating the overall execution time as measured by a clock on the wall. This is a problem not only for computation, but for all kinds of human endeavors that demand coordinated labor. The solution, as can be found all around us in society, is to have a small number of entities, in our case CPU cores, perform the serial labor at the same time that the bulk of the work is carried forward by a great many more such entities. Finding ways to keep all entities usefully busy boils down to a complicated process of planning and coordination. Our scientific simulation program must involve such planning and dynamic coordination at multiple levels of the computing system to avoid the computation repeatedly stalling. It turns out that this problem of code design is not difficult in principle, but it is very complex and can take a great deal of testing and effort to get right. As computing systems become ever more complex, this part of the code development process takes up a greater and greater portion of the total coding and testing time. Nevertheless, it is essential to running successfully on very large and powerful systems. One of the challenges of the NSCI, embodied in its fourth objective, is to develop a software ecosystem that eases this task. Doing that can have a dramatic impact on the pace of scientific progress.

Developing a Division of Labor

To address Amdahl's law successfully, we must identify the portions of the overall work that are inherently slow to execute but do not constitute a significant fraction of the total. Then we must identify portions of the code that will perform this work and package that work into separate processes that run alongside all the other processes in a parallel execution model. Some of these processes are provided by the computing system's supporting software and hardware, and the others we must provide as part of our running application. Examples of provided software are the I/O and message passing utilities. These execute alongside our program, sometimes on the same hardware, but often on separate systems, such as network interface cards and disk array controllers. Even so, we must take considerable care to use them in such a way that we are never held up, idling while waiting on these tasks to complete. Examples of software that we write ourselves to perform slow but small parts of the work are code for the preparation of output data, and in particular, global radial profiles, extensive imagery, and highly compressed, quantitative 3D data. In each case, whether we provide the code that performs the task ourselves or not, we must decide what fraction of our 14,000 nodes to devote to these tasks, and we must carefully design the rest of the code so that we can begin to perform this special work long before any other part of the ongoing computation requires it to be completed. This is a general technique. For our stellar hydrodynamic problems, the slow parts of the computation represent so little aggregate work that we can assign more than enough nodes to this portion of the whole, so that we are assured that it will always be done in time. Even doing this, our specialized processes, which we think of as administrative processes, are so small in number that they represent an overhead of only 6 percent.

We divide our 14,000 nodes into teams, and for each team we assign a team leader process and a team timekeeper process. When our job starts, all our MPI processes are created, and the very first thing they do is decide whether they are a worker, a team timekeeper, or a team leader. Next, they divide up all the work and assign portions to themselves that will minimize the chance that they will ever have to wait on input from another process. For the simulation in Figures 1–4, we had 512 teams, with 216 workers and two administrators per team. This resulted in a very low administrative overhead of less than 1 percent, but still our workers essentially never waited on others. Our most recent code design uses teams of 32 workers with these same two administrators. To keep these more numerous administrators busy, we are in the process of designing new administrative work for them to perform, such as nucleosynthesis postprocessing. The team timekeepers are specialized to perform global reductions, and since these can be time critical, we never ask them to do anything else. The results of these reductions are always used only when they are slightly out-of-date, so that no one needs to wait upon their arrival. The team leaders, like team leaders in real life, write the reports. We provide enough teams so that this work never fills all their time, and therefore they never hold up the overall execution of the job. We divide the work of preparing the output data for our reports between workers and team leaders, so that team leaders are mostly collating the reports rather than thinking about them. Thus, for example, each worker takes several pictures of its data by executing the srend volume rendering package,[14] but we have the team leaders composite these pictures together into single images. We tried sending the data to the team leaders and having them take the pictures instead, but then they were unable to get their work done before the next output dump was due to be produced. Although our workers take these pictures, we have them do this at their convenience, in moments when all their participating threads are available, so that their other work is never held up unduly by this output data preparation. They also check only occasionally to see if their team leader is ready to receive this data, and only send it on request. This avoids overwhelming the team leaders with data contributions and allows the team leaders to process their worker's data incrementally and at a comfortable pace, sending it out to disk in large globs that enable the I/O subsystem to perform at its best.

Dealing with Inherently Slow Delivery of Necessary Data

Another inherently slow process that requires careful coordination and planning is data acquisition. This slow process must be carefully scripted at multiple levels of the computing system's memory hierarchy. For data that must be acquired from other nodes of the system over its interconnect, we must request its transmission long before we need to use it. This requires careful coordination, because it cannot be sent to us unless it has already been generated at the remote node. It is less often recognized that similar coordination and planning are required for data acquisition from our own node's memory. Just as with remote node data acquisition, this requires careful bundling of the data so that it can be acquired in a small set of large globs rather than many tiny parcels. One must also plan ahead, so that the data can be requested long before it is needed. This need for planning data acquisition on the same node where a worker process executes is becoming increasingly acute today. Our approach to this problem, which has been described in detail elsewhere,[13,18,19] is illustrated graphically in Figure 5. We have each thread of a worker process perform a dynamically scheduled series of independent grid pencil updates. Each grid pencil is a strip of briquettes, aligned in the direction of the present 1D pass of our variant[15,16] of the PPM gas dynamics algorithm[17] and extending all the way through the present grid brick being updated on a node of the machine. We acquire the next grid briquette, a tiny cube of grid cells four on a side, while updating the present one and while writing back the updated result for the previous one. This methodology causes all the data needed for the update to be resident in the CPU core's on-chip cache. It also causes all this data to be packaged as 32-word, perfectly aligned vectors, as indicated in Figure 5. The CPU core can process the 32 words in each vector all at once, on some equipment, or in no more than four operations.

The measure of the success of all this algorithm and code design effort,[20] of course, is the ultimate overall performance.[18,19] For the code we ran to produce the results in Figures 1–4, this is roughly 11 percent of the advertised peak performance of the 13,952 nodes of Blue Waters that we used. Had we crammed this calculation onto one-eighth this number of nodes, our tests reveal that we could have run at 12 percent of the peak performance, which is a small additional benefit to receive for waiting eight times longer for the computation to complete. By running our code on so many nodes of a large, tightly interconnected machine like Blue Waters at NCSA, we are able to perform a single run of millions of time-step updates in a single week. At our lab at the University of Minnesota, we have built clusters of PC workstations with numbers of such nodes ranging from 8 to 24. The aggregate delivered performance to our codes was about 1 Tflop/s, while a run on Blue Waters like that shown in Figures 1–4 achieves roughly 440 times this performance. This is the difference between a day and a year to get a job done, or, as we mentioned earlier, about three times the difference between walking and flying in a jet plane.

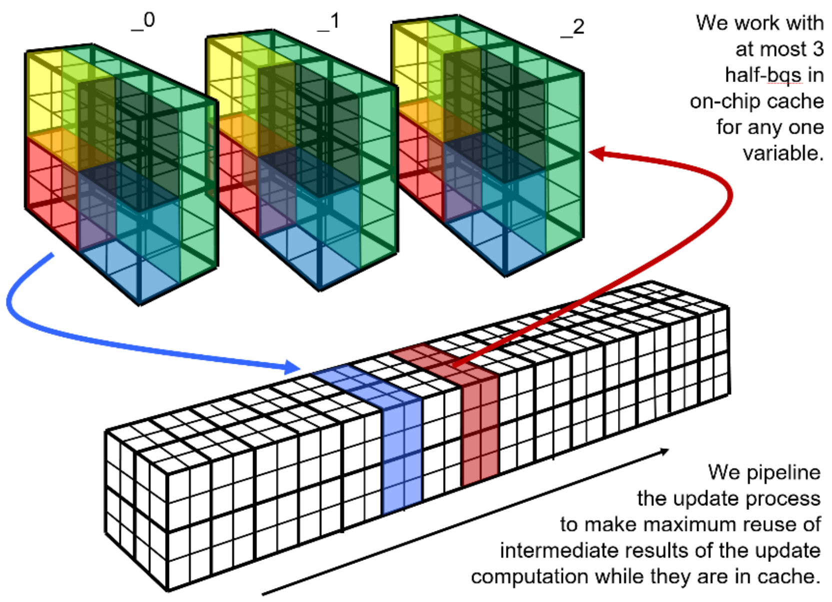
_0
_1
_2
We work with at most 3 half-bqs in on-chip cache for any one variable.
We pipeline the update process to make maximum reuse of intermediate results of the update computation while they are in cache.

Figure 5. A diagrammatic representation of the grid pencil update.

ACKNOWLEDGMENTS

Simulations and code development discussed in this article were performed on the Blue Waters machine at NCSA under support from NSF PRAC awards 1440025, 1515792, and 1713200, as well as a subcontract from the Blue Waters project at NCSA. Partial support for the code development also came from contracts from the Los Alamos and Sandia National Laboratories. The scientific research described here was supported by the NSF through CDS&E grant AST-1413548 and through subcontracts from NSF's JINA-CEE center (PHY-1430152). Simulations were also carried out on Canada's Westgrid resources and research supported by an NSERC Discovery grant. Srend development was partly supported by NASA under award number NNX15AQ02A.

REFERENCES

1. F. Herwig, "Evolution and Yields of Extremely Metal-poor Intermediate-Mass Stars," *Astrophysical J. Suppl.*, vol. 155, 2004, pp. 651–666.
2. Jones et al., "H Ingestion into He-Burning Convection Zones in Super-AGB Stellar Models as a Potential Site for Intermediate Neutron-Density Nucleosynthesis," *Monthly Notices of the Royal Astronomical Society*, vol. 455, 2016, pp. 3848–3863.
3. C. Ritter et al., "Convective-Reactive Nucleosynthesis of K, Sc, Cl, and P-Process Isotopes in O-C Shell Mergers," *Monthly Notices of the Royal Astronomical Society*, vol. 474, 2018, pp. L1–L6.
4. O. Clarkson, F. Herwig, and M. Pignatari, "Pop III i-Process Nucleosynthesis and the Elemental Abundances of SMSS J0313-6708 and the Most Iron-Poor Stars," *Monthly Notices of the Royal Astronomical Society*, vol. 474, 2018, pp. L37–L41.
5. N. Iwamoto et al., "Flash-Driven Convective Mixing in Low-Mass, Metal-deficient Asymptotic Giant Branch Stars: A New Paradigm for Lithium Enrichment and a Possible s-Process," *Astrophysical J.*, vol. 602, 2004, pp. 377–387.

6. S.W. Campbell, M. Lugaro, and A. . Karakas, “Evolution and Nucleosynthesis of Extremely Metal-Poor and Metal-Free Low- and Intermediate-Mass Stars,” *Astronomy and Astrophysics*, vol. 522, 2010, p. L6.
7. F. Herwig, “Evolution of Asymptotic Giant Branch Stars,” *Ann. Rev. Astronomy and Astrophysics*, vol. 43, 2005, pp. 435–479.
8. F. Herwig et al., “Convective-reactive Proton-12C Combustion in Sakurai's Object (V4334 Sagittarii) and Implications for the Evolution and Yields from the First Generations of Stars,” *Astrophysical J.*, vol. 727, 2011, pp. 89–103.
9. B. Freytag, H.-G. Ludwig, and M. Steffen, “Hydrodynamical Models of Stellar Convection,” *Astronomy and Astrophysics*, vol. 313, 1996, pp. 497–516.
10. s. Jones et al., “Idealized Hydrodynamic Simulations of Turbulent Oxygen-Burning Shell Convection in 4π Geometry,” *Monthly Notices of the Royal Astronomical Society*, vol. 465, 2017, pp. 2991–3010.
11. H. C. Spruit, “The Growth of Helium-Burning Cores,” *Astronomy and Astrophysics*, vol. 582, 2015, p. L2.
12. F. Herwig et al., “Global Non-Spherical Oscillations in 3-D 4π Simulations of the H-Ingestion Flash,” *Astrophysical J. Letters*, vol. 792, 2014, p. L3.
13. P. R. Woodward et al., “Simulating Turbulent Mixing from Richtmyer-Meshkov and Rayleigh-Taylor Instabilities in Converging Geometries using Moving Cartesian Grids,” *Proc. NECDC2012, LA-UR-13-20949* (NECDC 12), 2013; http://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-13-20949.
14. T. Wetherbee et al., “In-Core Volume Rendering for Cartesian Grid Fluid Dynamics Simulations,” *Proc. XSEDE Conf.: Scientific Advancements Enabled by Enhanced Cyberinfrastructure* (XSEDE 15), 2015; doi.org/10.1145/2792745.2792780.
15. P.R. Woodward, “The PPM Compressible Gas Dynamics Scheme,” *Implicit Large Eddy Simulation: Computing Turbulent Fluid Dynamics*, F. Grinstein, ed., Cambridge University Press, 2007; www.lcse.umn.edu/ILES.
16. P.R. Woodward, F. Herwig, and P.-H. Lin, “Hydrodynamic Simulations of H Entrainment at the Top of He-Shell Flash Convection,” *Astrophysical J.*, vol. 798, 2015, pp. 49–74.
17. P. Colella and P.R. Woodward, “The Piecewise-Parabolic Method (PPM) for Gas Dynamical Simulations,” *J. Computational Physics*, vol. 54, 1984, pp. 174–201.
18. P.R. Woodward et al., “Scaling the Multifluid PPM Code on Blue Waters and Intel MIC,” *Proc. Extreme Scaling Workshop*, 2013; www.xsede.org/documents/271087/586927/Woodward.pdf.
19. P.-H. Lin and P.R. Woodward, “Transforming the Multifluid PPM Algorithm to Run on GPUs,” *J. Parallel and Distributed Computing*, vol. 93, 2016, pp. 56–65.
20. P.R. Woodward, J. Jayaraj, and R. Barrett, “mPPM Viewed as a Co-Design Effort,” *Proc. 1st Int'l. Workshop on Hardware-Software Co-Design for HPC* (Co-HPC 14), 2014, pp. 33–40.

ABOUT THE AUTHORS

Paul R. Woodward is a professor at the University of Minnesota, where he has directed the Laboratory for Computational Science & Engineering (LCSE) since its founding in 1995. His research interests include numerical simulation of fluid dynamics, scientific visualization, and the astrophysics of stars and galaxies. Woodward received a PhD in physics from the University of California, Berkeley. He received IEEE’s Sidney Fernbach award in large-scale computation, and together with a team of several others, the Gordon Bell award in performance. Contact him at paul@lcse.umn.edu.

Falk Herwig is a professor at the University of Victoria. His research interests include the formation and evolution of the elements in stars and galaxies. Herwig received a PhD in Astrophysics from the Astrophysical Institute Potsdam (AIP)/University Kiel. Contact him at fherwig@uvic.ca.

Ted Wetherbee is an instructor in mathematics and computer science at Fond du Lac Tribal and Community College. His research interests include scientific visualization and multi-physics applications for integrated education and local research activities. Wetherbee received an MS in mathematics from Ohio University. Contact him at ted@fdltcc.edu.