

# An Integrated Framework for Interactive Simulations

Manish Parashar

Department of Electrical and Computer Engineering, Rutgers University, 94 Brett Road, Piscataway, NJ 08854

## Abstract

This paper describes the design of an integrated framework for interactive, parallel and distributed scientific simulations. The framework is driven by a unified computational engine that can “seamlessly” integrate parallel/distributed computing, interactive visualization, and scientific databases. The fundamental innovation is a unifying data-representation and an associated semantically specialized distributed virtual shared memory that can simultaneously support dynamic computations, visualization, and interaction operations. We believe that such a unified framework will enable realistic simulations offering the prospect of dramatic insights into complex phenomenon.

## I INTRODUCTION

The next generation simulations of complex physical phenomenon will be built on top of a unified computational engine integrating parallel and distributed computing, real-time interactive and collaborative visualization, and scientific databases. The simulations will exploit advances in these technologies to provide dramatic insights into complex systems such as interacting black holes and neutron stars, formations of galaxies, seismic models of the whole earth and geophysical models of reservoirs and aquifers. This paper presents the design of an integrated framework that can provide such a computational engine. The key innovation of the framework is a *unified distributed and dynamic data-management substrate* that can fundamentally integrate parallel adaptive computations, observation, experimentation, and interactive visualization and analysis.

The primary data-structure implemented by the data-management substrate is a virtual *hierarchical distributed and dynamic array (vHDDA)*. vHDDA extends array access semantics to dynamic, heterogeneous, and physically distributed data objects spanning different storage types/ vHDDA objects encapsulate distribution, communications, coordination, and load balancing. The objects are then used drive large-scale distributed adaptive applications based on a family adaptive solution techniques such as adaptive finite-differences, hp-adaptive finite elements, and adaptive fast multipole methods. These techniques seek to improve the accuracy of the solution by dynamically refining the computational grid in regions of high local solution error. Parallel and distributed implementations of these techniques lead to interesting problems in dynamic data-distribution and load-balancing, communications and coordination, and resource management.

Visualization, analysis and interaction operations have been fundamentally integrated into the vHDDA-based computational engine so that they can be “seamlessly” integrated with the simulation. Visualization and analysis have been traditionally viewed as separate (typically post-processing) phases of the simulation process. Consequently visualization techniques have been independently parallelized with their own data-structures and distributions, and computational data is fed to them using files. These approaches require redundant effort and can be expensive. By enabling the visualization and analysis operations to execute on the same distributed data-management engine as the computation (i.e. vHDDA), we can provide a “*window in the oven*” view of an ongoing simulation. Similarly sensors and actuators for interaction are directly integrated into the computational data objects to enable interactive experimentation and steering. The result is a transformation from traditional “batch” simulations to more efficient interactive ones.

vHDDA data layout and storage mechanisms are based on extendible hashing techniques that have been traditionally applied to manage large dynamic databases. These mechanisms can be naturally extended to allow data layout in the persistent storage to exploit the nature and structure of the computations, and enable computational in-core data-structures to directly interact with this data. Such an extension will enable the definition of a uniform application programming interface where the data access semantics are independent of its physical location – whether in-core or in a persistent store.

We believe that such a unified framework will enable realistic simulations offering the prospect of dramatic insights into complex phenomenon. The rest of this paper outlines the design of the vHDDA data-structures and the data-management substrate.

## II AN INTEGRATED FRAMEWORK FOR INTERACTIVE SIMULATIONS

A block diagram of the proposed computational infrastructure is shown in Figure 1. It is founded on a unified data-management substrate that transparently spans distributed processor memories as well as disks and persistent storage systems. The primary component of the data-management scheme is a natural mapping from a multidimensional, multiscale application domain to 1-dimensional storage that is directly derived from the application domain coordinate system. This mapping enables implementation of a “*semantically specialized*” virtual shared memory where the memory address space is derived from application domain coordinates. As a result

heterogeneous computational and data objects can be directly accessed from virtual memory based on their location in the computational domain rather than using conventional memory addresses and pointers. The resulting data-structure implements a virtual, Hierarchical, Distributed and Dynamic Array (vHDDA) spanning multiple levels of the distributed memory hierarchy. It extends array semantics to hierarchical, dynamic and physically distributed objects, providing a uniform application programming interface that transparently span heterogeneous data and storage types. vHDDA is now used to drive parallel adaptive computations involving multiple scales and structures, advanced visualization and analysis operations including feature extraction and feature tracking, and collaborative interaction and steering. The resulting computational infrastructure can seamlessly integrate computations with observations, experimentation and analysis.

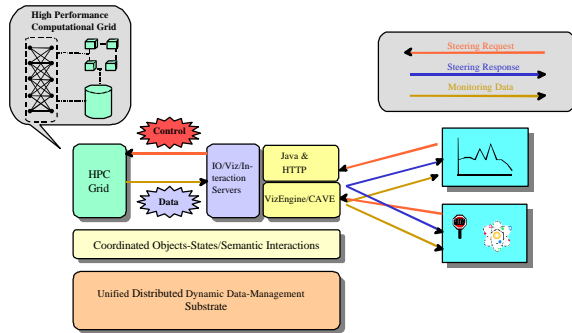


Figure 1 – Integrated Framework for Interactive Simulations

### III UNIFIED DATA-MANAGEMENT SUBSTRATE

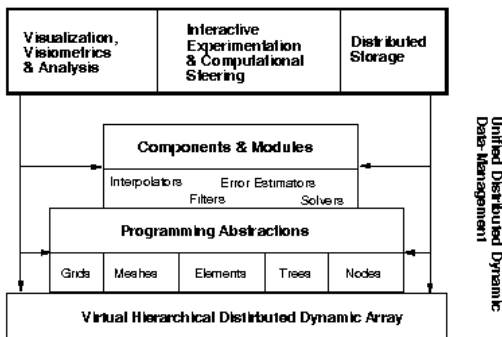


Figure 2 – Unified Data-Management Substrate

Figure 2 shows a schematic overview of the unified data-management substrate. The lowest layer of the infrastructure implements the vHDDA. The next layer of the infrastructure adds application semantics to vHDDA objects to implement application data-types such as grids,

meshes and trees. This layer provides object-oriented programming abstractions[6] that can be used to directly implement advanced computation, visualization, and analysis operations. The upper layers of the infrastructure provide components and modules for computation, visualization, and interaction. vHDDA construction is described below.

#### A Virtual Hierarchical Distributed Dynamic Array (vHDDA)

The primitive data structure provided by the data-management substrate is a virtual, hierarchical, distributed and dynamic array providing uniform array access to heterogeneous dynamic objects spanning distributed address spaces and multiple storage types. The array is hierarchical in that each element of the array can be an array; it is dynamic in that the array can grow and shrink at run-time. Communication, synchronization and consistency of vHDDA objects are transparently managed for the user. The lowest level of the array hierarchy is an object of arbitrary size and structure. A key motivation for defining such a generalized array data-structure is that most application domain algorithms are formulated as operations on grids and their implementation defined as operations on arrays. Such array-based formulations have proven to be simple, intuitive and efficient, and are extensively optimized by current compilers. Furthermore providing an array interface to the dynamic data-structures allows implementations of new parallel and adaptive algorithms to reuse existing kernels at each level of the vHDDA hierarchy. A key feature of the vHDDA is its ability to preserve application locality requirements despite its distribution and dynamics. The vHDDA design is based on the application of the fundamental system engineering design principle of Separation of Concerns[3]. Applying Separation of Concerns to a convention array data-structure, it can be decomposed into an ordered index-space, storage corresponding to each index in the index-space, and accesses mechanism that enable retrieval of data objects associated with an index. Similarly, vHDDA is composed of (1) a hierarchical index-space and (2) associated distributed dynamic storage and access mechanisms.

##### 1 Hierarchical Index Space

Hierarchical index spaces are derived directly from the application domain using space-filling[8]. These are computationally efficient, recursive mappings from N-dimensional space to 1-dimensional space. The solution space is first partitioned into segments. Space-filling curve then passes through the midpoints of these segments. A 2-dimensional mapping hierarchy is shown in Figure 3. Space filling mappings encode application domain locality and maintain this locality through expansion and contraction. This implies that points that are local in 1-dimensional space are mappings of points that are local in N-dimensional space. Furthermore, the self-similar or recursive nature of these mappings can be exploited to map

hierarchical domains and to maintain locality across different levels of the hierarchy. The index-space is used by vHDDA as its virtual shared memory address space to assign storage to application domain objects. It is used for partitioning the application domain, for resolving names and addresses, and for communication coordination. Figure 4 shows the derivation of a hierarchical index space from a 2-dimensional adaptive grid.

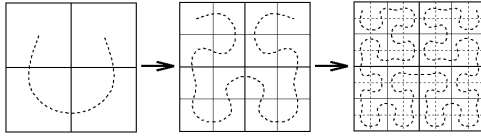


Figure 3 – Recursive Space-filling Mapping

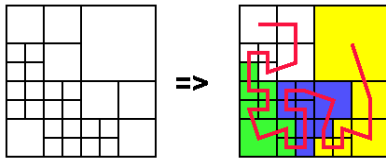


Figure 4 – Hierarchical Index-Space Generation

## 2 Distributed Dynamic Storage and Access

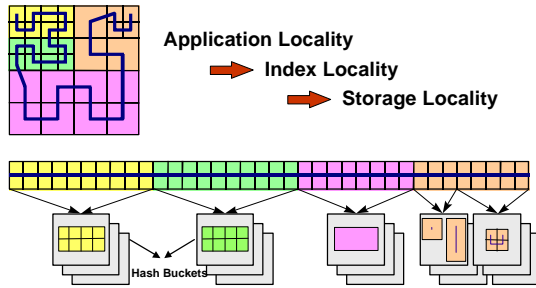


Figure 5 – Distributed Dynamic Storage and Access

Data storage is implemented using extendible hashing technique[2] with contractions of the hierarchical index space as the hash key space. Extendible hashing provides efficient data-management mechanisms for dynamic databases. Spans of the hash keys space are mapped to units of storage called hash buckets, and expansion/contraction of the key space is handled efficiently by splitting and merging these buckets. These operations are local and involve at most two buckets. vHDDA buckets can span multiple storage types. As spans of the index space are mapped to contiguous storage within a bucket, index locality is translated into storage locality. Data locality is preserved without copying. vHDDA buckets store multiple data types associated with each index, and can span multiple levels of the virtual memory hierarchy. vHDDA is distributed by partitioning its index space among processors and assigning ownership to vHDDA buckets. Buckets are also used as the unit of communication and caching. The overall vHDDA storage scheme is shown in Figure 5[4].

## B Integrating Visualization & Analysis

Visualizing and analyzing multistructure data from evolving 3D distributed adaptive simulations is non-trivial due to the irregular nature and immense amount of data to be processed and understood. Multistructure includes multiresolution, multiblock, structured and unstructured datasets. Multiresolution grids, which underlie adaptive simulations, allow regions of the grid to be locally refined. Datasets associated with these grids contain multiple levels of refinement, and tracking and visualizing the changes in features between refinement levels and across scales is crucial. A 3-dimensional multiresolution grid hierarchy is shown in Figure 6. Multiblock grids are used to describe highly irregular domains as unions of regular “block” grids, and are similar to hybrid grids (see Figure 7).

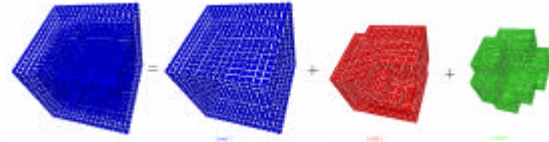


Figure 6 – 3D Multiresolution Grid Structure

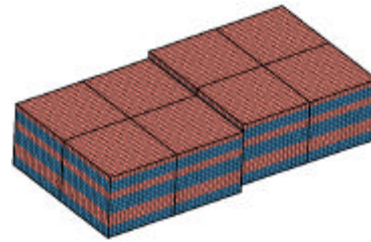


Figure 7 – 3D Multiblock Grid Structure

The vHDDA-based data-management substrate integrates visualization and analysis with computations by using vHDDA base objects to directly drive the visualization and analysis operations. The basic data-management requirement for implementing parallel visualization techniques for multistructure grids is the definition of a hierarchical structure that can maintain object relationships across space, time and resolution, and efficiently translate these relationships into access locality in distributed memory space. The vHDDA hierarchical index spaces encode these relationships so that visualization objects that are close in geometric space and/or resolution result in indices that are close in the derived index space. This proximity is then translated to storage locality by the extendible hashing mechanism. A vHDDA-based linked-list structure can thus be created by associating each visualization object in the multistructure dataset with a unique index in the index space (based on its geometric location) and using these indices as “smart pointers” in the linked list. Each object in the list maintains indices of related objects and can access these objects by using them to index into the vHDDA. Since operations on the lists of connected component require nearest-neighbor traversals, these operations can be achieved with minimal remote accesses (since vHDDA buckets maintain index locality). Similarly, coalescing operations for creating

isosurfaces and for computing overlaps can be performed efficiently using logarithmic tree-based algorithms.

Visualization and interaction interfaces are developed by customized AVS[1] modules using vHDDA data-objects. AVS is a component-based visualization and programming system. This approach enables computation and visualization methods to directly communicate with other modules at run-time, and utilize their capabilities for rendering, quantification and analysis. Since these modules and the computation use a common data-structure, data can be directly streamed to AVS without any serialization and data reformatting. Interaction and steering activities are also built into the same modules and use the same data-structure.

### C Integrating Computations with Data

The vHDDA based data-management substrate provides a natural means for integrating computations with observational and experimental data by enabling objects in memory and in persistent mass storage systems to be directly associated using their relationship in the application domain. For example, computed data-objects and observed values at a particular point in the domain can be directly associated. This association is due to the space-filling curve derivation of the hierarchical index space which is used by the vHDDA to map the applications domain to physical storage. As vHDDA storage buckets can span all levels of the virtual shared memory hierarchy, and its array access interface is independent of the object's storage location, we can define a unified virtual storage system for computational, observational and experimental data that exploits the nature and extent of the application. Data corresponding with a particular region in the computational domain (and hence a span of the index-space) can now be directly and simply associated with the corresponding observational data in a scientific database (also mapped to the same index-space).

### D Interactive Experimentation and Steering

Applying the design principal of "separation of concerns", vHDDA base objects can be directly extended with interactive visualization and steering capabilities[7]. This is achieved by allowing the vHDDA base objects to inherit from the visualization and interaction abstract classes. These classes encapsulate visualization, interaction and input/output operations and define base methods and operators. When application objects are constructed by specializing the vHDDA base objects, these methods and operators are also specialized. For example, a grid implemented by specializing vHDDA objects will also specialize visualization, interaction and input/output operators to grids and grid functions. These objects will then directly generate connectivity information for displaying the grid structure. They will also generate grid interaction sensors to define and interpret possible interactions with the grid structure such as refinement, coarsening, error estimation, interpolation and input/output.

## IV CONCLUSIONS

In this paper we presented the design of an integrated framework for the next generation of interactive and immersive scientific simulations. The key innovation of the design is a *unified distributed and dynamic data-management substrate* that can fundamentally integrate parallel adaptive computations, observational and experimental data, and interactive visualization and analysis. The design presented was driven by applications requirements from scientific and engineering domains including numerical relativity and astrophysics (black hole and neutron star interactions), geophysics (seismic whole-earth models) and petroleum engineering (oil-reservoir simulations), and will in turn enable research advancements in these domains. We believe that such a unified framework will enable realistic simulations offering the prospect of dramatic insights into complex phenomenon.

## V REFERENCES

- [1] Advanced Visual Systems (AVS). AVS Inc. <http://www.avs.com>.
- [2] H.F. Korth, A. Silberschatz, "Database System Concepts," McGraw Hill. New York, 1991.
- [3] M. Parashar and J.C. Browne, "System Engineering for High Performance Computing: The HDDA/DAGH Infrastructure for Implementation of Parallel Structures Adaptive Mesh Refinement," IMA Volumes in Mathematics and its Applications, Springer-Verlag, 1997.
- [4] M. Parashar, J.C. Browne, "Distributed Dynamic Data-Structures for Parallel Adaptive Mesh Refinement," Proceedings for the International Conference on High Performance Computing, India, December, 1995, 22-27.
- [5] Parashar, M., and Browne, J.C., "On Partitioning Dynamic Adaptive Grid Hierarchies," Proceedings of the 29<sup>th</sup> Annual Hawaii International Conference on System Sciences, Hawaii, January 1996, 604-613.
- [6] Parashar, M., and Browne, J.C. Object-Oriented Programming Abstractions for Parallel Adaptive Mesh-Refinement. In proceedings of Parallel Object-Oriented Methods and Applications (POOMA), Santa Fe, NM. February 1996.
- [7] Parashar, M., and Browne, J.C. Integrated Data-Management for Computational Steering. Accepted for publications to the 31<sup>st</sup> Annual Hawaii International Conference on System Sciences.
- [8] Samet, H. The Design and Analysis of Spatial Data Structure. Addison-Wesley, Reading, Mass, 1989.