From Large Simulations to Interactive Numerical Laboratories

Alexander S. Szalay
The Johns Hopkins University, Baltimore, MD 21218, USA
szalay@jhu.edu

Abstract

High Performance Computing is becoming an instrument in its own right. The largest simulations performed on our supercomputers are now approaching petabytes. As the volume of these simulations is growing, it is becoming harder to access, analyze and visualize these data. At the same time for a broad community buy-in we need to provide public access to some of the simulation results. This is becoming another Big Data challenge, where we have to move the analyses and visualizations right where the data is. The paper will discuss the challenges in creating such interactive numerical laboratories.

1 Introduction: Data in HPC

1.1 Motivation

The nature of science is changing, it is increasingly limited by our ability to analyze the large amounts of complex data generated by our instruments and simulations: Jim Gray's "Fourth Paradigm" of science [3, 14]. The largest numerical simulations use tens of millions of hours of CPU time, yet most of the analysis must be performed while the simulations are running, since the output data are too large to be moved or stored for reuse. As a result, it is difficult, if not impossible, to confront model predictions with the exponentially increasing amounts of observational data. Scientists in many disciplines would like to compare the results of their experiments to data emerging from numerical simulations based on first principles. This requires not only that we run sophisticated simulations and models, but that the results of these simulations are available publicly, through an easy-to-use portal. We have to turn the simulations into open numerical laboratories where anyone can perform their own experiments.

Integrating and comparing experiments to simulations is a non-trivial data management challenge. Not every data set from these simulations has the same lifecycle. Some results are just transient and need to be stored for a short period to analyze, while others will become community references, with a useful lifetime of a decade or more. As we have learned over the years, once the data volume reaches a certain size, we have to move the analysis to the data [24, 25] rather than the traditional approach, which moved the data where our computers were. With these large data volumes we have to approach the data in a fully algorithmic fashion — manual exploration in small files is no longer feasible.

1.2 The emerging simulation data challenge

There is an ongoing effort world-wide to build an exascale computer by approximately 2018. This reflects a scale-up by a factor of 30 compared to the top machines in late 2013. Some of the largest particle-based simulations today already exceed a trillion particles. If we only store their positions and velocities, and a single particle identifier, we have to save 56TB for each snapshot, uncompressed. Needless to say, that this in itself represents a challenge.

At the same time, fewer and fewer codes scale to run well on millions of cores, and as a result, fewer and fewer people will use these ever larger systems. There will be an increasing gap between the wide science community and these top users. It is increasingly important to create usable science products that can be used by a much broader pool of users; otherwise community support will be soon endangered. Thus, we postulate that it is extremely important to identify a mechanism through which data products from the largest simulations in science can be released, publicly shared and used, potentially over extended periods.

1.3 Data lifecycles of simulations

On the Fly Analysis

Most truly large numerical simulations are analyzed on the fly. The analysis tools are integrated with the simulation, and the derived data products are computed while the simulation is running. As these quantities represent only a small fraction of the data, it is easy to save these values often to disk. Full restart checkpoints are thus only generated quite infrequently. The disadvantage is that if a new analysis idea emerges after the run is finished, the whole simulation needs to be redone.

Private reuse

Sometimes a few tens of snapshots are saved to scratch disks, tightly coupled to the supercomputers, and a occasionally a segment of the simulation can be regenerated later from restarting from the nearest snapshot. This is typically done by the same team who ran the original simulation.

Public reuse

There are selected cases when the simulation outputs are made available, usually done through sharing the limited number of snapshot files. These are typically placed at a public file server, and can be downloaded at will. However, this model of data sharing poses practical limits to data downloads at a few terabytes at most. The limiting factor is usually the network bandwidth, although the available storage at the user's end is also a problem.

Public Service portal

Sometimes the simulation outputs are made available through publicly available services, enabling the users to perform either some extractions or computations over the data. This idea of "virtual data" has been around for more than a decade, but it has found limited uses. The Earth Science community has used OpenDAP [19] to expose large data sets and enable a RESTful URL to subset and aggregate the data. In astrophysics, the Millennium Database [16] has been the forerunner of such efforts. In this scenario, the creation of the public service portal and its complex functionalities requires a substantial effort, thus it is only worth doing if the data set will remain public for an extended period, at least a few years.

Archival and long-term curation

There are very few simulations that have reached this stage of their lifecycle. Here an extra challenge is that not every simulation will be equally used by the public, and over the years; some of them will fade into irrelevancy while others emerge as a community reference. It is these latter simulations which need to be kept for a long time, even if just for comparison and reference. For such collections, used by many different refereed publications, reproducibility of these analyses will become another reason to keep the data, even when better and higher resolution alternatives become available. However, as the price of both storage and computation are expected to follow the current trends and become cheaper every day, these "legacy" data sets will comfortably fit in the shadow of the latest and best simulations.

1.4 Petascale numerical laboratories

Even though the largest simulations today are approaching hundreds of billions of particles or grid points, the size of the output generated is typically a few tens of TB, rarely exceeding 100TB and almost never reaching a PB. As already mentioned above, there are many reasons for this. The larger the computer, the more cumbersome checkpointing becomes, and while the biggest machines have a Tbit/sec aggregate sequential bandwidth to their storage, copying 100TB will still take 800 seconds, too long to do often, limiting the number of snapshots. As the interconnect speeds are not going to increase by a factor of 30-100, it is likely that this limitation remains in place, thus even once we have exascale machines, the outputs will still remain in the few PB range.

When the primary consideration is checkpoint restart, it is enough to have a small number of snapshots. On the other hand, if the goal is to be able to reconstruct the fine-grained spatial and temporal history of the simulation, and look at any part in detail, it is important to match the high spatial resolution with an appropriate temporal resolution, i.e. a large number of snapshots. For example, if we simulate a Milky Way-like galaxy, and want to study its dynamics in detail in our laboratory, we need to save outputs more frequently than the characteristic rotational period of 10^8 years. This means that even the simulations need to be designed and run differently if the target is to create a long-lived numerical laboratory used by hundreds of scientists.

2 The Challenges

The premise of this paper is that due to community pressure and demand for repeated posterior analyses some of the best and largest simulations will be turned into public numerical laboratories. This process presents nontrivial challenges for every step of the simulation and reuse process. In particular, we need to figure out

1. Where to store the data for the reuse
2. How to move the data to this location
3. How to look at it, how to render peta/exabytes
4. How to interface to the data
5. What analysis patterns will be used
6. What architectures to use for the posterior analysis

In the following subsections we will look at these in some detail.

2.1 Moving the data

Storage attached to supercomputers has a premium. In order to minimize the overhead due to checkpointing, this storage has to be very fast, also on the fast interconnect. The primary function of this storage is to dump the contents of the RAM into large sequential files at maximum speed. To keep petabytes of data for months to years and running interactive analyses by hundreds of users requires a different architecture.

The most important aspect of this is that the long term data storage should use inexpensive storage technology so that it can keep accumulating important data. This facility (we will call it Numerical Laboratory from now on) has to be interconnected with a high speed network to the primary data site, the HPC machine. Today, many large data sets are moved via "sneakernet" [11], i.e. copying the data onto disk drives and shipping or carrying the drives physically. This works for data that fits onto a small number of hard disks, up to about 10TB, or 3 disks. Beyond this, mounting tens of disks, partitioning, and copying the data, and keeping track of what is where becomes impractical.

Network transfer of larger files is also non-trivial. Over a 10 Gbps link (assuming wire speed) we can move a 100TB data set in about 80,000 sec, or 1 day. Over such a link realistic transfer speeds are more in the 6Gbps/sec range. It is important to note that these transfers must be done from disk-to-disk. Both ends of the transfer must have fast enough I/O to support 1GBps disk reads and writes. Once transfers take more than a few hours, often they get interrupted. Luckily, user friendly technologies, like Globus OnLine [10] are emerging which support automatic recovery from such errors. These are maximizing the network throughput by enabling parallel striping of the transfer.

Many of the national supercomputer sites are in the process of upgrading their connections to 100G. Both ESNet and Internet2 are moving their backbones to 100G. Using these, one can theoretically move a PB in a day. Internet2 provides Layer 3 10G services, but for a while the 100G connectivity will remain Layer2 only, making the routing more cumbersome.

Local networking rules and firewalls can have a dramatic impact on the transfer speeds. Many institutions are setting up demilitarized zones (DMZs) to isolate a segment of their network that can talk to the outside at high speeds. OpenFlow [20] is emerging as a mechanism to have applications configure and negotiate fast virtual circuits across the continent. So, while it is still rather painful today to move data exceeding 100TB, this capability is expected to become increasingly available to a wider set of institutions, while PB-scale data transfers will still be relatively limited.

2.2 Interfacing to the data: files or services?

The simplest thing is to simply provide access to the simulation files. This is the most frequent mechanism used today to disseminate outputs. However, these large files store the data in large contiguous chunks, usually partitioned following the domain decomposition, in a packed (often proprietary) binary format. Access to the data this way is only useful if the simulations are small, or all the data is loaded back into memory. If each snapshot is tens of TB, there are not many user-owned computational facilities that can support such an amount of RAM. Also, the user of the data has to understand the data formats, units, organization, domian decomposition and boundary conditions. Often mastering these requires several days, in not a week, of preparation and coding, prohibitive for a casual user.

Having a service oriented access with a finer granularity has a lot of advantages. Data can be provided "shrink-wrapped", with calibrations applied, in proper physical units, like the archive for the Sloan Digital Sky Survey [12]. Aggregates can be computed and subsets can be extracted on demand. The result is that the user performs the low level data processing right on top of the data, inside the Laboratory, reducing the volume to be transmitted across the wide area network. Of course, this requires enough computing power tightly integrated with the data storage. On the other hand, from a community perspective the individual users can run their petascale analyses form a laptop, rather than forced to build their own facility, so this may be a reasonable value proposition for the community as a whole.

2.3 Access patterns for analysis

There are several distinct access patterns how a posterior analysis will access the simulation data.

Global analytics

These are analyses that require access to the whole simulation volume. Examples of these are computing the Fourier transform of a scalar field, like density, spatial correlation functions, or computing the density for a particle-based simulation using a smoothing kernel. These operations are quite similar to those performed on a supercomputer during the simulation itself (Poisson solvers, etc). As the global data access touches a very large fraction of the data, this pattern is optimally satisfied by loading the sequential dumps of the snapshots. As data sizes grow, running these analyses over a heavily partitioned parallel system is creating an increasing communication overhead, e.g. the transpositions needed for the 3D FFT.

Rendering

Many other analyses are similar to a rendering of a subvolume. These of course include visualizations of large subsets. In cosmology in particular, an interesting challenge emerges: as we create a rendering of a light cone in the Universe, create an image of what a virtual telescope would see, we have to account for the finite speed of light within the simulation. As we look farther and farther along the line of sight, we see simulation at an earlier time. This means that unless we have many snapshots stored, such renderings are impossible to recreate.

The effects of gravitational lensing is currently computed by collapsing the distribution of mass along the line of sight within the simulation onto idealized slabs, where the projected mass density acts as a refractive index, to scatter light gravitationally. This is really a computer graphics problem, rendering the mass onto multiple planes and then doing a parallel ray-tracing.

Another problem resembling rendering is computing the pair annihilation of dark matter particles in a simulation. The collision rate of particles is proportional to the density squared, and the cross section can have an additional dependence on the velocity of the particles. Thus, the resulting maps can be very sensitive to the local environment. However, this is still a rendering problem, where we take a scalar field attached to a particle based support and create a projected map.

Typically these patterns require accessing a substantial fraction, but not all, of the simulation volume. Various spatial data structures, like octrees, space filling curves, can help in optimizing the amount of data to be read into RAM. A spatial index with a flexible geometric search capability is required for efficient data access. GPUs can be extremely useful in these computations.

Localized access

There are computational tasks which require a very localized access. For example in order to compute the fluid velocity at a given location, we need to extract a small volume of a few grid cells in each direction, the size of our kernel, and then compute an interpolated value. Such immersive computations have been used to create a smart laboratory for turbulence [27].

In cosmology simulations often we run the simulation with collisionless dark matter only. Then a friend-of-friends algorithm is used to group of particles into halos and subhalos. Some of these will be called galaxies. Using the merger history of these groups one can heuristically estimate what the physical properties of these galaxies would be. In the Millennium simulation this is precomputed, However, there is a growing need to be able to perform these computations on-demand, changing the heuristic rules leading to a galaxy and understand their impact on the final ensemble of observable galaxies.

These patterns all require a very fine-grained localized access to small parts of the data. Here databases with a high-resolution spatial index can make a huge difference in enabling fast data reads.

2.4 Immersive access, virtual sensors

For a scalable analysis we need to come up with a data access abstraction or metaphor that is inherently scalable. For the user it should not matter whether the data in the laboratory is a terabyte or a petabyte. The casual user should be able to perform very light-weight analyses, without downloading much software or data. Accessing data through the flat files violates this principle: the user cannot do anything until a very large file has been physically transferred.

On the other hand one can create a so-called immersive environment, where the users can insert immersive virtual sensors into the simulation [18]. These sensors can then feed data back to the user. They can provide a one-time measurement, they can be pinned to a physical (Eulerian) location or they can “go with the flow as comoving Lagrangian particles. In this case, assuming that the sensors can access the data server side very fast, the only scaling is related to the number of particles.

By placing the particles in different geometric configurations, users can accommodate a wide variety of spatial and temporal access patterns. The sensors can feed back data on multiple channels, measuring different fields in the simulation. They can have a variety of operators, like computing the Hessian or Laplacian of a field, or applying various filters and clipping thresholds.

This pattern also enables the users to run time backwards, impossible in a direct simulation involving dissipation. Imagine that the snapshots are saved frequently enough that one can interpolate particle velocities smoothly enough. Sensors can back-track their original trajectory and one can see where they came from, all the way back to the initial conditions. We can also imagine cases where one simply retrieves the velocity of the sensor particles, and applies a special equation of motion involving other factors (inertia, friction, stochastic perturbations) and move the particles externally, possibly on a users laptop, anywhere in the world. This simple interface can provide a very flexible, yet powerful way to do science with large data sets from anywhere in the world.

2.5 Data organization, streams and indexes

Much of the efficiency of the numerical laboratory will depend on how the data is organized and laid out on the storage devices. If we talk about several petabytes of data, the only real storage option is using an array of hard disks. These are still mechanical device. They speed of revolution has remained largely constant over the last decade, and the inexpensive commodity drives have settled at around 7200 rpm. As the density of disk platters grows, the disk capacity increases by the inverse square of the magnetic domain size. The read speed increases as the inverse, thus it takes longer and longer to read the ever larger disk platters, disks are becoming sequential devices.

For large data volumes we need first of all a maximally sequential layout. As many numerical simulations cover a 3+1D domain, simply storing a large subvolume as a separate file, ordered in x,y,z is less than optimal for small localized accesses: for each small subvolume we need to perform many separate reads, spaced by a stride apart, for a single extraction. This results in a very poor cache localization, and many random accesses on the external disks.

Space filling curves have been used for a long time [22] as a cache resilient way to access such spatial data. Indeed, using a Peano-Hilbert or a Morton curve enables a mapping of a 3D region to a contiguous sequential location, providing a handle to maximally sequential data access. The key of the space filling curve can then be used as an index into the physical data store, and every subvolume can be represented as a set of range queries over the key. Such operations are highly optimized in relational databases, if this is mapped onto a clustered index (primary key).

At the same time, in particle-based simulations the need often arises to follow trajectories of individual particles. In this case we need an inverted index. Rather than storing the whole date over again, a rather expensive proposition for petabyte data sets, one can create highly compressed indexes, which use the fact that particles move with a finite velocity [6]. This way we can represent the whole trajectory of the particles in a fraction of the original storage.

2.6 Visualization and Rendering

A visual exploration of the data is a powerful tool of science. While we have immersive caves and large video walls with tens of millions of pixels near our supercomputers, they are not helping much a wider community. For many scientists the most useful aspect of visualization is to help them decide whether an idea or hypothesis is to be discarded or worth further exploration. The faster data can be transformed and looked at, the faster the whole feedback loop from the conception of a new idea to transformation and rendering can be performed, the more useful it becomes. Today much of the user-side visualization is done with a few open source toolkits, executed near the user, typically on a workstation with a mid to high end GPU. This approach, while it can give a fast turnaround, is unfortunately not scalable to large data volumes.

Fast GPUs have changed the landscape for scientific visualization. Their rendering speeds are such, that most visualization projects are limited by the bandwith for data to be loaded into GPU memory. For this reason, just like for Big Data analytics where we have to execute our workflows as close to the data as possible, large scale visualization must also be performed right on top of the data [26].

For petabytes, only remotely driven visualizations will have the bandwidth to access the underlying data sets. This approach is aided by the fact that we have reached a point when user devices can reliably access streaming content at speeds allowing HD video almost anywhere. In this case we only have to move the result of the rendering, not the source, reducing data volumes by many orders of magnitude.

The data access patterns necessary for petascale visualizations have several algorithmic and architectural consequences. The visualizations must be done in parallel, where the distributed components are co-located with a distributed data storage. At the same time, visualizations can have very different locality requirements. In some cases we need the data at a small granularity, where indexes based on space filling curves work very well. In other cases we need a large fraction of the subvolume, organized in (x,y,z) order. Given the ultra high speeds of the GPU on-board RAM, reorganizing data in GPU memory is much faster than regular RAM, thus loading the data in memory in (almost) arbitrary order, and then copying into its final location on the GPU works extremely well in practice [5].

3 Architectures of Numerical Laboratories

One might first think that given the recent proliferation of commercial cloud solutions will bring an immediate solution to the table. There are several interesting and important aspects of these cloud architectures. The most important is that they do not try to be everything for everybody. Due to their commercial nature they have been built with a razor-sharp focus, with several major tradeoffs in their design.

These are extreme scalability, economies of scale and resilience. The clouds built by Amazon, Google and Microsoft are all based on the principle that components will fail, and systems have to survive and recover, and data cannot be lost. This is accomplished by massive data replication and excessive monitoring. The commercial clouds must also be cheap to operate.

The data in these clouds is largely unstructured, from web content to click streams. Also, the connectivity of the data is quite different: mostly it is large graphs describing links or social networks. Much of the software framework for the processing reflects this necessity.

Scientific data is quite different. Much of our data, especially the simulations is highly structured, with well defined schemas and data structures. There are well-defined 2 and 3D domains, with distinct timesteps, and boundaries. With the large data volumes, there are similarities as well, especially how in science we are also exploring phenomenological correlations rather than only searching for strict, causal relationships. Nevertheless, due to the more structured nature of scientific data, the computations can utilize vectorizable architectures, like GPUs much better.

The economics is also different. Pricing for the external users of the commercial clouds is driven by the perceived business model of a small, but dynamically growing business, which is generating revenue from efficiently using data analytics over small to mid-scale data volumes. The prices for compute cycles are generally very competitive when compared to the costs of running your own.

The same cannot be said about storage, once it reaches Terabytes. The monthly cost of a TB of storage on the commercial clouds is roughly equal to the price of a TB disk drive [21]. Of course, in the cloud one gets redundant storage, service guarantees and the costs of operations are included. But there is also a download charge, and access costs apply. So, for many science projects today putting data in a commercial cloud on a pay as you go basis is simply not affordable. Also, bandwidth between the data and compute is marginal. As a result, very few mid to large scale data sets have been placed into the cloud, other than a few experiments, or making use of occasional free storage space by the providers. It is important to note, that this is entirely a function of the pricing should cloud storage prices drop by order of magnitude, for mid-size data sets in the few tens of TB this could be a very attractive proposition.

For data sets in the 100 TB to PB range, getting data into the cloud is hard. The costs per year are close to 1M, and structured access to the data is not as fast as one can achieve with a dedicated system.

3.1 Amdahl number

It is good to have a simple characterization of the I/O needs of both computational tasks and the underlying computer architectures. Bell, Gray and Szalay [2] has introduced the Amdahl number, the sequential I/O speed in bits/sec divided by the total instructions/sec, following Amdahls seminal paper [1]. This simple metric shows a clear distinction between supercomputers, with Amdahl numbers around 0.001, workstations at around 0.1, and data analytics engines at Amdahl numbers close to 1.

Furthermore, the Amdahl number can also be computed for a computational task, like a numerical simulation, or their posterior analyses, by dividing the output size by the CPU hours, converted to the appropriate units. Typically, our supercomputer simulations have Amdahl numbers much lower than the hardware they are running on. This is almost by definition, as one could only saturate the hardware by continuously maxing out the I/O, at the expense of computations, defeating the purpose of using a supercomputer.

On the other hand, posterior analyses of the data can have very different Amdahl numbers. Streaming analyses of petabytes can easily require Amdahl numbers close to unity. Small-granularity database queries are in this category. As outlined in Section 2.3, there is a whole spectrum of computational tasks whose needs cover a wide range of Amdahl numbers.

One can also define an approximate Amdahl number for a GPU. As an example, an NVIDIA Kepler K20 card has about 1.3Tflops and its I/O is limited by the practical bandwidth of 6GB/sec over a PCIe gen2 backplane. This translates to an Amdahl number of 0.037, definitely higher than a supercomputer.

3.2 Database servers or supercomputers?

The challenge is to define an architecture that is (i) efficient and inexpensive for storing petabytes of data, (ii) provides streaming high bandwidth to the data, (iii) can occasionally perform compute-intensive tasks involving large amounts of data. It is clear, that supercomputers have enormous processing power, but they are too expensive for the posterior processing of large data sets. It is also clear that database servers have been successful in providing fast, indexed access to small granularity data, they do not have the compute power to perform some of the more floating point intensive analyses. But between these two edge cases there is a continuum of problems to be tackled successfully.

3.3 Storage hierarchies for data analysis

Given the wide range of data access patterns, we need a system that is capable of accessing data both sequentially at high speeds, but can also provide high IOPS for small accesses, and is capable of high computational performance. The challenge is that building a homogeneous system that satisfies all these criteria would be next to impossible, and certainly very expensive.

The key is heterogeneity. One can build systems that can be close to optimal along each of these requirements individually. We know how to build fast streaming engines that can move data sequentially at backplane speeds from Hard disks all the way to GPUs, like the JHU Data-Scope. Also we know how to use solid state disks to provide close to a million IOPS per server. And we are learning how to use large memory servers.

Commodity interconnect is becoming very affordable. Infiniband and 40G Ethernet are below 500 USD per port, and latencies are approaching a microsecond. With a million IOPS from an SSD system, the latency of reading a packet of data is similar than over our fastest networks. For memory intensive problems one can combine commodity systems soon with 6TB of memory, tens of TB of SSD and ultra fast interconnects, blurring the traditional domain decomposition issues.

4 Use cases and examples

4.1 Turbulence

Over the last few years we have built several different numerical laboratories that have been used to prototype and explore the ideas outlined in this paper. The first of these deals with turbulence. We took a relatively small 1024^3 box containing a simulation of isotropic turbulence. The simulation consisted of approximately 10,000 timesteps. We created snapshots of the first 100 timesteps, then every 10th, until we reached 1000 snapshots. The outputs were the three velocity components and the pressure. The total data volume is about 30TB.

The data has been reorganized into small, 8^3 cells, which are labeled by a Morton curve or z-index. These cubes are stored as BLOBs in a relational database system (Microsoft SQL Server). The z-index is used as a partitioning key. The database is used as an indexed data access layer. The BLOBs contain a user-defined data type (UDT) of a multidimensional array of a single homogeneous primitive data type. Arrays can be either small (8kB) or large (up to 2GB). The UDT has various methods enabling Matlab-like array manipulations [8].

On top of the indexing is a generalized spatial search library that can be used for data extractions defined by various geometric primitives and their Boolean combinations [17]. All these functionalities heavily use the object-relational interface of SQL Server, enabled through the CLR integration [13].

The data is accessible through various web services. The first service we have implemented was the one of the virtual sensors. Users can use their own laptop to insert an array of sensors, defined by a 3+1 dimensional coordinate. The user can lay out these sensors in various geometric patterns, providing a simple yet flexible interface for accessing data. The service returns the interpolated velocities at the different locations. Other derived quantities, like computing the velocity gradient tensor can also be requested. There are several different language bindings available (C++, Fortran, MATLAB).

The users can then run various statistical analyses over the data returned. They can also implement their own path integrator, and use the velocities to move the particles, but with the additional capability of adding additional components, like a small stochastic force term. However, using the database, one can do analyses that would not be possible in real time when the simulation is running on the supercomputer.

As turbulence is a dissipative phenomenon, one cannot run the simulation backward. As we have stored the full temporal history of the velocity field, at a high enough resolution for path interpolation, we can move the particles backwards in time, and determine where all the particles in a given neighborhood come from! This has led recently to a paper in Nature [9].

This simple interface has been highly successful, by 2011 we have exceeded 100 billion sensor lookups delivered to users all over the world. With time, it became clear, that moving many massless Lagrangian particles along their trajectory was a pattern frequently requested. Thus, we added another service were you can place the sensors which then would move with the flow, reporting their positions and velocities.

For the isotropic turbulence database the data set is large enough that it is non-trivial to stream the data from a remote server to a visualization engine. We have performed several experiments in visualization tight on top of the data. We have built a separate server with a fast motherboard with multiple PCIe x16 slots (gen 2). The system also contained a high-end visualization card (NVIDIA GTX 690) and a high throughput LSI 9211 disk controller, connected to 120GB OCZ Vertex 3 SSD drives, running Windows Server 2008. We have copied 32 snapshots of the turbulence database over to this system. The disks were able to deliver over 1GB/sec sequential throughput.

A rendering engine built in DirectX 10 was used to create various visualizations. The renderer was sending queries to the database, which in turn delivered the velocity field in 8^3 blocks, together with a z-index used for the database organization. The arrival order of the blocks was not guaranteed. The data was immediately copied into the GPU. The real 3D position of each block was calculated from the z-index and the high bandwidth of the GPU memory enabled a very fast copy of the data into its final location, organized in the usual (x,y,z) order for optimal rendering.

The visualization engine had options for calculating various derived quantities, like the velocity gradient tensor, and its invariants, implement various smoothing operators for multiscale visualizations, or to visualize velocity fields by inserting clouds of particles which then moved along the streamlines [5]. During these experiments we have achieved over a GB/sec data transfer speeds, mostly limited by the number of SSDs.

4.2 Cosmological N-body simulations

The Millennium Simulation

We are currently in the process of building several different numerical laboratories out of large-scale cosmological N-body simulations. The first of those was based upon the the Millennium simulation [23]. Its database, built by Gerard Lemson [16] has been in use for over 7 years, and has hundreds of regular users, and has resulted in several hundred refereed publications. The database contains value added data from a simulation originally only containing 10B dark matter particles. These particles form hierarchical clusters, so called sub-halos and halos, which in turn become the sites where galaxies might form. A semi-analytical recipe was used to create mock galaxies in the simulations, and their hierarchical mergers were tracked in a database structure. The merger history was used to assign a plausible star formation rate to each galaxy which in turn can be used to derive observable physical properties.

The database contains several such semi-analytic scenarios. The set-oriented SQL query language makes it remarkably easy to formulate very complex aggregate queries over the temporal history of various subsets of galaxies and create samples that can be compared directly to observations.

Via Lactea II and Silver River

These simulations seek to simulate the formation and evolution of the Milky Way galaxy. Via Lactea II [7, 15] was the first of the two, with about 1B dark matter particles, it was one of the highest resolution such simulation. It led to a bigger simulation, the Silver River, which has over 50B particles.

There are several non-trivial numerical experiments that can be turned into a public laboratory. The first involves a computation of the annihilation of the dark matter particles. The Fermi satellite is in the process of observing high energy radiation from space, and it is not unreasonable to expect that in the near future it might see a sign of excess radiation from the Galactic Center that can be attributed to the annihilation of the elusive dark matter.

In preparation, we have built a numerical experiment, where users can interactively choose a viewpoint relative to the Galaxy, and then compute an image of the high energy radiation coming from the annihilation. Furthermore, the users can choose the physical properties of the cross section for the annihilation. Different physical models for the dark matter can yield dramatically different images. Originally, computing a single annihilation map took over 8 hours on a regular server. Using a rendering algorithm written in the shader language of Open GL, and streaming the data efficiently from the disks, we are now able to compute a single map in 23 seconds, changing the kinds of analyses one can perform [28]. Soon a public service enabling scientists to play with such scenarios will be live.

One can imagine immersive uses of the numerical laboratory, similar to the turbulence case. The Sloan Digital Sky Survey has detected the remnants of several dwarf galaxies which collided with the Milky Way, and their structure was disrupted in the collision, their stars ended up in an elongated “stream [4]. One can easily image a scenario, where a user can take a small dwarf galaxy, consisting of about 1M particles, picks a trajectory and shoots it into the Milky Way. The dwarf galaxy moves in the time-dependent, high resolution potential of the original simulation, and the user can watch interactively how the particles are scattered into the elongated stream, and repeat the experiment until the desired final state is achieved. Our group is currently in the process of building such an interactive application.

The Indra suite of simulations

In astrophysics we have a single universe to observe, and we cannot even change our position, thus we are restricted to a single realization of space surrounding us. The only way to estimate the statistical uncertainty arising from the fact that we can only observe a finite sample of the Universe, the so called “cosmic variance is to use simulations. Most of the large simulations are trying to be the largest of their kind, thus focusing all the computational resources in generating a single realization.

There is also a need to create a statistical ensemble of simulations. Instead of focusing on the largest number of particles, or the highest resolution, we should create an ensemble of simulations, which can be used to estimate statistical uncertainty for our various observables, which can then be used to assess the significance of real observations of the Universe.

We are in the process of creating the Indra suite, eventually consisting of 512 simulations of 1B particles each. While none of the individual simulations are particularly large, their ensemble will have an data volume of more than 1PB [6]. Currently we have already ran the first 100 realizations, and have more than 200TB of data. Using this ensemble one can study covariances on such large scales, where none of the simulations have enough independent subvolumes. As it turns out, studying the density fluctuations on scales of the so called Baryon Acoustic Oscillations (BAO) is one of the most important challenges of cosmology today, and Indra will uniquely be able to support such explorations.

5 Summary

The next generation simulations will have trillions of particles, petabytes of output. Some of the simulations will be multigrid, some will be a combination of fluids and discrete particles. A wide community of users is ready to use the output of these simulations, and compare them to experiments and observations. The science community has used several of these prototype numerical laboratories successfully, even artfully. It is clear that the stage is set to have a new platform, which bridges the current gap between supercomputers and database servers. It is clear that the key to efficient data analysis is a good data layout and efficient data access.

The data access patterns identified in this paper are quite diverse, there is no single platform or architecture that fits all. Most likely we will need a heterogeneous ecosystem of system components, each optimized for different data access patterns and enough internal bandwidth to move parts of the data to the appropriate system dynamically. This way we can get close to optimal layout for most analyses. In order to do so, we have to understand and characterize how different data access and analysis patterns map onto different system architectures and components.

6 Acknowledgements

The author would like to acknowledge support from the National Science Foundation OIA-1124403, ACI-1244820, OCI-1040114 and the Gordon and Betty Moore Foundation, grant 109285. Many of the ideas emerging here have grown out of conversations, joint projects and many years of collaborative effort with a lot of collaborators, without whom much if this would not have happened. I would like to thank Jim Gray, Jose Blakeley, Tamas Budavari, Kai Buerger, Randal Burns, Laszlo Dobos, Greg Eyink, Ed Givelberg, Kalin Kanov, Gerard Lemson, Piero Madau, Charles Meneveau, Eric Perlman, Joe Silk, Tamas Szalay, Ani Thakar, Ethan Vishniac, Sue Werner, Ruediger Westermann, Simon White and Rosemary Wyse.

References

[1] G. Amdahl. Computer architecture and amdahl's law. *IEEE Solid State Circuits Society News*, 12(3):4–9, 2007.

[2] G. Bell, J. Gray, and A. Szalay. Petascale Computational Systems: Balanced Cyber-Infrastructure in a Data-Centric World. *IEEE Computer*, 39:110–113, 2006.

[3] G. Bell, T. Hey, and A. Szalay. Beyond the data deluge. *Science*, 323(5919):1297–1298, 2009.

[4] V. Belokurov, D. B. Zucker, N. W. Evans, G. Gilmore, S. Vidrih, D. M. Bramich, H. J. Newberg, R. F. G. Wyse, M. J. Irwin, M. Fellhauer, P. C. Hewett, N. A. Walton, M. I. Wilkinson, N. Cole, B. Yanny, C. M. Rockosi, T. C. Beers, E. F. Bell, J. Brinkmann, Ž. Ivezić, and R. Lupton. The Field of Streams: Sagittarius and Its Siblings. *The Astrophysical Journal Letters*, 642:L137–L140, May 2006.

[5] K. Bürger, M. Treib, R. Westermann, S. Werner, C. C. Lalescu, A. S. Szalay, C. Meneveau, and G. L. Eyink. Vortices within vortices: hierarchical nature of vortex tubes in turbulence. `http://arxiv.org/abs/1210.3325`, 2012.

[6] D. Crankshaw, R. C. Burns, B. Falck, T. Budavari, A. S. Szalay, and J. Wang. Inverted indices for particle tracking in petascale cosmological simulations. In *SSDBM*, page 25, 2013.

[7] J. Diemand, M. Kuhlen, and P. Madau. Formation and Evolution of Galaxy Dark Matter Halos and Their Substructure. *The Astrophysical Journal*, 667(2):859–877, Oct. 2007.

[8] L. Dobos, A. S. Szalay, J. A. Blakeley, T. Budavári, I. Csabai, D. Tomic, M. Milovanovic, M. Tintor and A. Jovanovic. Array requirements for scientific applications and an implementation for Microsoft SQL Server. EDBT/ICDT Array Databases Workshop, 13-19, 2011.

[9] G. Eyink, E. Vishniac, C. Lalescu, H. Aluie, K. Kanov, K. Bürger, R. Burns, C. Meneveau, A. S. Szalay. Flux-freezing breakdown in high-conductivity magnetohydrodynamic turbulence. Nature, em 497,466, 2013.

[10] Globus online website. `https://www.globus.org`.

[11] J. Gray, W. Chong, T. Barclay, A. S. Szalay, and J. VandenBerg. Terascale sneakernet: Using inexpensive disks for backup, archiving, and data exchange. *CoRR*, cs.NI/0208011, 2002.

[12] J. Gray, A. S. Szalay, A. R. Thakar, P. Z. Kunszt, C. Stoughton, D. Slutz, and J. vandenBerg. Data Mining the SDSS SkyServer Database. *ArXiv Computer Science e-prints*, Feb. 2002.

[13] J. Gray, A. S. Szalay and G. Fekete. Using Table Valued Functions in SQL Server 2005 To Implement a Spatial Data Library. `http://arxiv.org/abs/cs/0701163`, 2007.

[14] T. Hey, S. Tansley, and K. Tolle. *The Fourth Paradigm Data-Intensive Scientic Discovery*. Microsoft Research, Redmond, WA, 2009.

[15] M. Kuhlen, P. Madau, and J. Silk. Exploring Dark Matter with Milky Way Substructure. *Science*, 325(5943):970–973, Aug. 2009.

[16] G. Lemson and t. Virgo Consortium. Halo and Galaxy Formation Histories from the Millennium Simulation: Public release of a VO-oriented and SQL-queryable database for studying the evolution of galaxies in the LambdaCDM cosmogony. *ArXiv Astrophysics e-prints*, Aug. 2006.

[17] G. Lemson, T. Budavári and A. S. Szalay. Implementing a General Spatial Indexing Library for Relational Databases of Large Numerical Simulations. Proc. SSDBM 11 Conference, 509-526, 2011.

[18] Y. Li, E. Perlman, M. Wan, Y. Yang, C. Meneveau, R. Burns, S. Chen, A. Szalay, and G. Eyink. A public turbulence database cluster and applications to study Lagrangian evolution of velocity increments in turbulence. *Journal of Turbulence*, 9:31–+, 2008.

[19] Opendap website. `http://docs.opendap.org/index.php/UserGuide`.

[20] OpenFlow Project website. `https://www.opennetworking.org/sdn-resources/onf-specifications/openflow`.

[21] How to build a cheap petabyte? `http://blog.backblaze.com/2009/09/01/petabytes-on-a-budget-how-to-build-cheap-cloud-storage`.

[22] H. Samet. *Applications of spatial data structures - computer graphics, image processing, and GIS*. Addison-Wesley, 1990.

[23] V. Springel, S. D. M. White, A. Jenkins, C. S. Frenk, N. Yoshida, L. Gao, J. Navarro, R. Thacker, D. Croton, J. Helly, J. A. Peacock, S. Cole, P. Thomas, H. Couchman, A. Evrard, J. Colberg, and F. Pearce. Simulations of the formation, evolution and clustering of galaxies and quasars. *Nature*, 435:629–636, June 2005.

[24] A. Szalay and J. Gray. The World-Wide Telescope. *Science*, 293:2037–2040, Sept. 2001.

[25] A. Szalay and J. Gray. 2020 Computing: Science in an exponential world. *Nature*, 440:413–414, Mar. 2006.

[26] M. Treib, K. Bürger, F. Reichl, C. Meneveau, A. S. Szalay, and R. Westermann. Turbulence visualization at the terascale on desktop pcs. *IEEE Trans. Vis. Comput. Graph.*, 18(12):2169–2177, 2012.

[27] Turbulence simulation services website. `http://turbulence.idies.jhu.edu`.

[28] L. Yang and A. Szalay. A GPU-Based Visualization Method for Computing Dark Matter Annihilation Signal. *Astronomical Data Analysis Software and Systems XXII*, 475:73, Oct. 2013.