Large Scale Cluster Surveys and Distributed Computing

James Annis

Experimental Astrophysics Group, Fermilab, Batavia, IL 60510

Abstract. The Sloan Digital Sky Survey is among the first of the new generation of large scale sky surveys that enabling the construction of revolutionary new cluster catalogs. I describe here some of the experiences of creating a SDSS cluster catalog, some of the computational challenges we encountered, and what we can expect in the near future.

ENTERING THE AGE OF TERABYTE SCALE SKY SURVEYS

There is only one sky: on it are 40,000 sq-degrees. Take an image of the sky with 1" pixels and you have 520 billion pixels. Call it 10 bytes/pixel and this image weighs in at 5 Terabytes. Astronomy is beginning to take these images and is moving to do so at a variety of wavelengths. Soon we will be doing it as a function of time. We will have access to huge amounts of high quality data, and we will be learning new techniques to analyze it all.

The Sloan Digital Sky Survey [1] is one of the first of the large scale surveys. It is an experiment to map 10,000 square degrees of the the Northern Celestial Hemisphere and to obtain spectra of a million of the brightest galaxies. It covers only one quarter of the sky, but it does so with 0.4" pixels and through 5 bandpasses. By the completion date of 2006, The SDSS will have have generated 15 Terabytes of images, 250 million objects strong object catalogs, and billion pixel scale spectral catalogs.

The first year data collection is complete and we now have experience with the analysis of terabyte scale astronomical data sets. What have we learned? The data volumes are large: a 200 sq-degree catalog is 12 gigabytes. The data rates are high: the data comes in at a rate of 200 sq-degrees a month. The data are complex: there are 120 parameters per object, a radial profile, 5 bandpass atlas images, and often detailed spectroscopy. And the data are reprocessed and re-calibrated on 6 month time scales, so is a moving target. All of these lessons will apply to the forth-coming surveys: the collection of astronomical data is riding the same Moore's Law wave that the analysis of astronomical data rides; and recalibration of data is notoriously fascinating to those who take and analyze the data because it is so useful.

CLUSTER SURVEYS IN THE TERABYTE AGE

Clusters of galaxies are useful tools for studying the growth of structure in the universe as well as being laboratories for the effects of environment on galaxy evolution. Using the large scale, high quality, and uniform data of the SDSS for finding clusters is a pleasure scientifically, but it does present certain technical problems that are generic to any cluster finding algorithm.

We can use the Chak-Bey[1] (Annis et al 2001) algorithm as an example. This algorithm takes a definite view on what constitutes a cluster: a luminous elliptical galaxy with quite uniform properties surrounded with a sequence of lower luminosity galaxies whose colors are remarkably similar to each other. It takes the view that in physical units clusters all look the same, modulo the number of galaxies on the sequence. Given that, one can predict with great precision their appearance at any given distance. In fact for any observed galaxy on the sky, one can ask if it could be that luminous central elliptical at any possible redshift. One maximizes the likelihood that it is by calculating the likelihood at all redshifts and choosing the maximum. This algorithm as a by-product produces a precise (1%) photometric redshift measurement.

Cluster finding is intrinsically an N^2 problem. The number of galaxies in the catalog is N. Checking the distance to all the galaxies surrounding it takes N^2 time. Doing so for every redshift z is then N^2R_z time, where R_z is the resolution in z, typically 10^2 to 10^3. Fighting the N^2 slowness involves minimizing N. Chak-Bey does so via a variety of heuristic ways. Primarily these take advantage of the natural tree structure of the data (imaging run, camera column, field number) to minimize N: clusters are only searched for in a given field and the surrounding 8

[1] Chak-Bey: red road in classic Mayan; the E/S0 ridgeline of clusters presents a red road to efficiently locating them.

fields, not the entire imaging run. Secondarily, one can use work at a given z for the entire field at a time, applying color cuts to bring down N before calculating the distances.

It is possible to transpose from the natural tree structure of the data into kd-trees built on position, magnitude and color. Kd-tree algorithms are $N \log N$ and are directly applicable to Chak-Bey in particular and cluster finding in general. There are other computer science developed algorithms of interest: k^{th} nearest neighbor searches and pyramid data models using multi-scale statistical summaries of underlying data points from immediately to mind. In the future of multiple Terabyte scale datasets, any algorithm that is not $N \log N$ may well be infeasible.

Bottlenecks

The complicated nature of our data in conjunction with our desire for standard data formats lead to our first bottleneck: I/O. Reading the FITS binary tables that contain the data on 200 sq-degrees of sky takes 14 hours. This corresponds to $1/3$ Mbyte/sec, clearly CPU dominated, traceable to a remarkably limited FITS reader. The solution is to make compact ASCII catalogs containing only the parameters of interest. Reading the compact catalog for 200 sq-degrees takes 5 minutes, 30 Mbytes/sec, and takes Chak-Bey back into the CPU dominated regime.

The second bottleneck is CPU time. Chak-Bey takes 40 CPU hours per 200 sq-degrees of sky on a 500 Mhz PIII CPU; taking the inherent photo-z aspect of the finder to its limits took this to 200 hours. Both are too long for an efficient code/run/debug cycle. Our solution to this is a cluster of commodity computers, now themselves commodity items thanks to the rise of the Internet economy. Observational astronomers are behind the curve in learning how to use clusters; fortunately we are fast studies. A modest cluster of 6 dual CPU compute nodes brings the compute time down to an acceptable 3.5 hrs.

Both these issues are made more problematic given the necessary calculation of the selection function[2]. The calculation is itself straightforward: add a cluster to the data and run the cluster finder to locate it, a large number of times. This is, of course, more stressful to the computational system than cluster finding is, by the ratio of $N * A_{test}/A_{total}$, where N is the total number of selection function tests run, A_{test} is the area of the sky used in the test, and A_{total} is the total area of the sky used in cluster finding. Since $A_{test}/A_{total} \approx 50$ and $N \approx 1000$, one sees the problem.

[2] The probability of selecting a cluster of given mass and at a given redshift

TURNING THE BOTTLENECK INTO A RIVER: DISTRIBUTED ARCHITECTURES

Distributed Cluster Finders

Locating quasars is not a spatial problem: one can decide on an object by object basis which ones are quasars. Cluster finding, on the other hand, is inherently spatial. At a minimum it is performed in the 2 dimensional space of celestial positions; often it has a third, redshift dimension. But cluster finding is only mildly spatial, as clusters have a limited extent. This is what makes the speedup of tree structure codes possible; it also makes possible the construction of distributed cluster finding codes.

The insight is the extent of the largest cluster one looks for. The angular extent depends both on the redshift and the bandwidth of the cluster finding filter. One looks for clusters in a given area; required to do so is a buffer area about the main area, of size equal to the largest extent. Chak-Bey typically uses 500 kpc; at $z = 0.1$ this comfortably fits in a single field. The 8 fields directly about the main field are used to ensure that any clusters at the edge of the main field are not missed. This is helped by the adaptive nature of Chak-Bey: the cluster likelihood measurement is made at the positions of galaxies, and the question of whether there is a cluster in this field devolves to the question of whether the central galaxy of the cluster is in this field.

The price paid for this ability to distribute is twofold. First, one must duplicate the data: the data in the buffer area will be duplicated at least once. This can be minimized by making the central area as large as convenient, and is in any case not a big problem. Second, one must have code to join the answers together at the end. In fact, this is an example of a hash machine, the computational analog of the hash-sort algorithm.

Datawolves: Specialized Clusters

The next step is to specialize the computing: build special purpose machines. Based on our experience with the analysis of terabyte scale astronomical data sets, we are rapidly converging on the Datawolf design: if Beowulfs are computational clusters optimized for efficient message passing parallel processing, Datawolves are clusters optimized for extremely large data sets.

The first design is for a database cluster: the SDSS science archive database SX (Szalay et al. 1999) has a master/slave architecture that allows a full, high speed scan of the entire data set to be distributed over the machines of a cluster. We expect aggregate throughputs approaching 1

Gbyte/sec. The results of the query are then sent out over TCP/IP to the querying source.

The second design is for an analysis cluster: the Terabyte Analysis Machine (Annis et al. 2000) is designed either to bring large amounts of CPU power to bear on the data stream from the database cluster, or, if the data are locally available, to do its own high speed data scan. In fact, we are exploring the possibilities of making database repartitioning and re-indexing available as a facility class instrument for high speed problem solving through advanced algorithms. We expect the general use of TAM to be for bringing raw compute power to bear on the SDSS dataset.

Distributed Computing

The vision of specialized clusters is closely associated with the vision of a grid of high-horsepower computing resources available to specialized users with specialize problems. For the most CPU bound problems, however, there is no real need for tightly bound clusters.

It may well turn out that there are many terabyte scale problems that are distributable enough that extremely loosely coupled clusters will be of great interest. We are thinking of the example of Seti@Home, or of the less celebrated but more useful approach of Condor[3]. Once one has the joining/hash machine code in place, there is no real reason to depend on the workload of a single cluster. We expect this to be an evolutionary pathway.

At this point, we have a virtual data engine. The galaxy catalogs form the starting point; the database cluster extracts the relevant galaxies and sends them on to the analysis cluster, where the cluster finding codes run, and the resultant cluster catalog is reported. To turn this into a truly on demand process will take significant work, a task being pursued by the GriPhyN collaboration[4].

Versioning

Inevitably, the virtual data engine approach brings up the pedestrian topic of versioning. Of course the science code must be versioned: quality science codes have a evolutionary life of their own that can extend a decade or more, and one *must* be able to go back to the code that produced a given catalog. Less obvious is the fact that the catalogs themselves must be versioned as both the finder code changes, and *as the input data changes*. Using a galaxy catalog, for instance, means that one has implicitly used choices of star/galaxy separation and limiting magnitude; as these change so will the cluster catalog. Even less obvious is the fact that cluster catalog creators will have to provide cross-catalog identification tables. The users of the catalogs will have spent considerable energy examining single objects; they have a right to know its true name in the next catalog, or at very least its fate.

THE FUTURE: MULTIPLE-SIGNATURE TECHNIQUES

A cluster is $\sim 10^{15} M_{\odot}$ mass in ~ 1 Mpc^3 volume. This is not directly observable, of course, but there are many observational signatures: the star light from galaxies is visible in optical/infrared imaging, the thermal bremsstrahlung from cluster plasma is visible via x-ray imaging, the inverse Compton scattering of cosmic microwave background photons by the cluster plasma electrons (the Sunyaev-Zeldovich effect) is seen via submm/radio imaging, and weak gravitational lensing induced distortion of the background galaxies by the cluster surface mass is visible by deep optical/infrared imaging. Next generation techniques will use as many of these signatures as are available. We are currently exploring techniques that place ROSAT All Sky Survey (Voges et al 1999) photons on an equal footing as SDSS galaxies for example, and are looking to incorporate weak lensing signals.

Given that in the future, the likelihood that any given place on the sky contains a cluster will be determined using galaxies, x-ray photons, SZ decrement pixels, and lensing signals in unison, we can expect to be developing cluster finder architectures that must connect to many database clusters to extract the data, and find many analysis machines upon which to run.

REFERENCES

1. York, D. et al., *Astronomical Journal*, 120, 1579 (2000)
2. Annis, J. et al., 2001. in preparation.
3. Annis, J., Garzoglio, G., Ruthsmandorfer, K., and Stoughton, C., *ACAT 2000*, these proceedings.
4. Szalay, A., Kunszt, P., Thakar, A., Gray, J., Slutz, D., and Brunner, R.J. 1999. *astro-ph/9912382*. (http://arXiv.org/abs/astro-ph/9912382)
5. Voges, W. et al. *Astronomy & Astrophysics*, 349, 389 (1999)

[3] http://www.cs.wisc.edu/condor
[4] http://www.griphyn.org