, , 1?? ()

Halo World: Tools for Parallel Cluster Finding in Astrophysical N-body Simulations

DAVID W. PFITZNER dwp@mso.anu.edu.au
Mount Stromlo Observatory, PB Weston Creek, Weston ACT 2611 Australia

JOHN K. SALMON johns@cacr.caltech.edu
THOMAS STERLING tron@cacr.caltech.edu
Center for Advanced Computing Research, Caltech 158-79, Pasadena, California 91125

Keywords: clustering, density estimation, parallel computation, astrophysics, N-body simulation

Abstract. Cosmological N-body simulations on parallel computers produce large datasets — gigabytes at each instant of simulated cosmological time, and hundreds of gigabytes over the course of a simulation. These large datasets require further analysis before they can be compared to astronomical observations. The "Halo World" tools include two methods for performing *halo finding*: identifying all of the gravitationally stable *clusters* in a point-sampled density field. One of these methods is a parallel implementation of the *friends of friends* (FOF) algorithm, widely used in the field of N-body cosmology. The new *IsoDen* method based on isodensity surfaces has been developed to overcome some of the shortcomings of FOF. Parallel processing is the only viable way of obtaining the necessary performance and storage capacity to carry out these analysis tasks. Ultimately, we must also plan to use disk storage as the only economically viable alternative for storing and manipulating such large data sets. Both IsoDen and friends of friends have been implemented on a variety of computer systems, with parallelism up to 512 processors, and successfully used to extract halos from simulations with up to 16.8 million particles.

1. Introduction

According to current cosmological theory, much of the mass in the universe is in the form of so-called *dark matter* whose only significant interaction is gravitational [7]. Estimates are complicated by a number of difficult-to-measure parameters, but it is likely that dark matter constitutes between 20% and 98% of the mass of the universe. During the evolution of the universe, this dark matter forms dense objects, called *halos*, due to gravitational instability. Within these halos, the dark matter is supported against further collapse by random motions. The normal matter in the universe collects at the centers of these halos, where star formation leads to the existence of luminous galaxies and other observable phenomena.

A detailed analytic understanding of the evolution of the dark matter is hampered by the highly non-linear nature of the problem, and the complexity of the structures formed. Hence numerical methods have become a very important tool for understanding this evolution, and for comparing cosmological theories with astronomical observations. In N-body simulations, a piece of the Universe is represented by a set of N discrete particles, which can be interpreted as a Monte Carlo sampling of the vastly more numerous dark matter particles. Each particle has a position, a velocity, and a mass. The particle masses are set at the initial time and remain fixed; the particle positions and velocities are numerically integrated forward in time under the influence of the gravitational field of all of the particles. Simulations with larger N are desirable because the sampling is more complete. Larger N also gives larger dynamic range, i.e., the ratio of largest to smallest scales which can be addressed in a single simulation. This is important, for example, for investigating small-scale structure in simulations with a volume large enough to sample a representative region of the universe [15], or for better resolution of substructure in simulations of clusters of galaxies [4].

The output of these N-body simulations is a list of the three-dimensional positions and velocities of all the bodies in the system. The raw data themselves do not provide direct physical insight — i.e., the individual bodies do not correspond to any particular object in the physical universe. Before drawing physical conclusions, the physically meaningful *halos* must be located and cataloged. Halos are stable, persistent, gravitationally bound collections of bodies that correspond to the locations at which galaxies or clusters of galaxies form. Halos may contain other halos in a hierarchy. This represents the idea that a small collection of bodies may be stable and gravitationally bound, e.g., the solar system, yet it may be contained within a much larger stable system, e.g., the Milky Way galaxy. (In practice, the hierarchies that form in current computer models have nowhere near the range of length and time scales of this illustrative example.) Once a list of halos is compiled, their properties, e.g., mass, location, velocity, shape, angular momentum, etc., may be compared statistically with those of observed galaxies. This is the standard methodology in cosmological simulations.

Halo finding is essentially a hierarchical clustering problem, where the number of clusters is very large and is not known in advance. As such, the techniques that are presented here in the context of astrophysical data reduction may find application in other fields where very large spatial data sets must be analyzed. Most of the techniques presented are not specific to astrophysical problems and are based on counting, neighbor-finding and density estimation, so they may have direct analogs in other problem domains. We also report on techniques that rely on problem-specific knowledge, e.g., the evaporation criterion discussed in section 3.2. While the specific physical concept of binding energy may be inappropriate for other disciplines, it illustrates the power of using domain-specific knowledge to craft appropriate statistical criteria.

Recently, systems with N larger than 322 million have been simulated on massively parallel computers [14]. These large simulations produce correspondingly large datasets, posing a challenge for analysis, which has traditionally been done on workstations. Practical considerations like available memory, reasonable turnaround time, and a desire to study time-dependent processes make it increasingly desirable for critical data analysis tasks also to run on parallel machines. The "Halo World" project is developing a set of parallel tools for detailed analysis of halo objects from very large N-body gravitational simulations, in preparation for correlation with future deep space object sky surveys such as the Sloan Digital Sky Survey (SDSS).

We expect the output of future simulations to approach terabytes, consisting of hundreds of "snapshots" each of several gigabytes.

The two methods presented here address the problem of halo finding. Halos coincide with peaks in the underlying density field, but not all peaks are stable. A density peak's stability depends on the velocities of its constituent particles. An individual particle is *gravitationally bound* to the density peak if its velocity is small enough that the particle is confined by the gravity of the density peak to remain in the spatial region of the peak. The density peak as a whole is gravitationally bound if enough particles are bound so that the density peak persists over time.

Strictly speaking, halos should be defined in the six-dimensional phase space that includes both velocities and positions. In practice, halos are identified as clusters in three-dimensional position space, and then are refined and possibly rejected if they do not meet the stability criterion in the six-dimensional phase space. In fact, the stability criterion itself is often never evaluated. Instead, peaks are accepted if they are sufficiently large, c.f., N_{min} in Section 3. This is justified by noting that in practice, most density peaks form by gravitational collapse, and as a result are automatically bound. The important exceptions are those that arise from noise fluctuations in the discrete sampling of the underlying distribution. These fluctuations will result in density peaks which are *not* bound, which we refer to as *spurious halos*. The size criterion implicitly relies on the fact that large chance fluctuations are extremely unlikely, so that any observed cluster above a certain size is almost certainly a genuine halo.

The complexity of the spatial, velocity and density structure of the data is illustrated in the left hand panels of Figure 1. The halos are the distinct clusters of points that are supported against gravity by the 'random' velocities of the particles. The magnitude of the problem is illustrated by Figure 5, which shows the results of halo finding in Model 1. The original data has 400 times as many particles. The number of particles in each halo ranges from ten up to tens of thousands. New simulation data sets are larger by a factor of about 20, placing proportionally larger demands on computational resources.

The Halo World research is developing its data analysis software to run on the most cost effective parallel computers available, clusters of commodity personal computers — "Piles-of-PCs" [2]. Although no more than the cost of a high-end scientific workstation, clustered PCs can retain the hundreds of gigabytes of data required for this problem in local secondary storage, and deliver in excess of a gigaflops sustained performance. The algorithmic approach to be taken is in part driven by two factors related to this class of system. Because the simulation data set is so large, it must be managed from secondary storage rather than held in main memory. Therefore, out-of-core solution methods will be employed for extracting halos from simulation data. They require the programmer or library designer to grapple with the long latencies and large block sizes associated with disk access — kilobytes of data are typically delivered in milliseconds, but smaller transfers cannot be made appreciably faster. Because performance of these systems is modest, even at a few gigaflops, an innovative algorithm is being implemented that mixes spatial and time domain extraction techniques.

Table 1. Simulation Parameters.

Model	N	Volume
Model 1	16,777,216	(250 Mpc)^3 cube
Model 2	525,002	(20 Mpc)^3 cube
Model 3	8,599	$\frac{4}{3}\pi$ (10 Mpc)^3 sphere

The halo finding methods have been applied to real data sets obtained from cosmological simulations. Table 1 lists the basic parameters of simulations which have been analyzed and will be referred to later. The Megaparsec is the conventional unit of distance in cosmology. 1 Mpc = 3.26 $\times 10^6$ light years. A 250Mpc cube contains about 5 million times the mass of our own Milky Way galaxy. Thus, the individual particles in model 1 are about one third the size of the Milky Way galaxy, and the halos that are extracted from the model will represent galaxies and clusters of galaxies ranging in mass from approximately that of our galaxy to several hundred times as large. Models 2 and 3 are sub-regions extracted from larger simulations. The results from Model 1 have been applied to the study of elliptical galaxies in [8].

In the next section we describe the computational platform on which Halo World executes. Then we describe the friends of friends (FOF) halo finding method in Section 3, and note some shortcomings. This motivates the subsequent description of the new IsoDen method in Section 4, based on kernel density estimation [11]. Section 5 explains how the two methods have been implemented on parallel computers. Section 6 presents timing results on a distributed memory parallel computer with up to 512 processors. Finally, Section 7 contains some specific plans for development of an out-of-core implementation and a mechanism for tracking halos in time without recomputing them from scratch.

2. Clustered PCs for Space Science Computation

A new class of computing system is emerging as an important low cost replacement for mid-level multiprocessors. Clustered PCs exploit the cost advantage of mass market manufacturing while retaining the performance benefits of leading edge fabrication facilities. The Beowulf project [2] has demonstrated that sixteen node systems can be assembled for less than \$50,000 that deliver performance levels, including operation rate, memory and disk capacity, and memory and disk bandwidth, within a factor of 2 of systems costing 20 times as much. The software environment is provided by the Linux system, a POSIX-compliant, freely available and widely used operating system. Interprocessor communication uses message-passing, accessed via a complete implementation of the standard MPI application programmer interface. Beowulf systems can make multi-gigaflops level performance available to a much broader community due to their low cost while providing a user interface familiar to computational scientists working with typical massively parallel processors. The Halo World project will use a Beowulf system to analyze N-body simulation data. This system consists of 59 Intel Pentium-Pro processors (200 MHz clock) for an aggregate system peak performance of 11.8 gigaflops, 7.5 gigabytes of main memory, and 182 gigabytes of internal secondary storage. Targeting the Beowulf class of systems will, however, not preclude use of commercial massively parallel processors since use of standard application programming interfaces and operating system services will permit easy porting of the tools to other MPI/Posix platforms. In fact, most of the results presented here were obtained on commercial systems like the Intel Paragon.

3. The Friends of Friends Method

In the friends of friends (FOF) method of halo finding [5], one specifies a linking length, h_{link}, and identifies all pairs of particles with a separation of h_{link} or less. Such pairs are designated *friends*, and halos are defined as sets of particles that are connected by one or more friendship relations, i.e., *friends of friends*. The linking length is usefully parameterized as a density, and following [12], we define a density threshold:

$$\rho_{min} = \frac{2\bar{m}}{\frac{4}{3}\pi h_{link}^3}, \tag{1}$$

where $\bar{m}$ is the average particle mass in the simulation. That is, ρ_{min} is the density of a sphere of radius h_{link} containing two average mass particles. In regions where the density is greater than ρ_{min}, particles will tend to be closer together than h_{link}, and will be linked together in the FOF method. Notice that the mechanics of the FOF method do not involve density, but the effect of the method is to identify density peaks above the threshold ρ_{min}.

A second parameter in FOF is the minimum number of particles, N_{min}, in a halo. The purpose of N_{min} is to reject spurious halos — i.e., groups of friends that do not form persistent objects in the simulation. Chance associations involving larger numbers of particles are less likely than those involving fewer, so by setting N_{min} sufficiently large one hopes to avoid most spurious halos.

Figures 1 and 2 show two sets of results, using $N_{min} = 10$ and $N_{min} = 30$, from FOF on Model 3. These figures demonstrate two problems with FOF: joining halos together, and poor distinction of small halos from noise. The first problem is that at the center of the cluster, FOF finds one large halo which is clearly composed of several distinct halos. This is because everything in a region where the density is above ρ_{min} is joined into a single halo, whether or not the region includes objects which are distinct at some higher density — a problem first noted by [3]. From the density plots it is seen that there is no value of ρ_{min} which will distinguish the halos in the high density region without missing some of the lower density halos.

The second problem is the arbitrariness and ineffectiveness of the N_{min} parameter. In our test, the value of $N_{min} = 10$ is too small, since many of the small halos found have high internal velocity scatter, and hence are not bound (c.f. figure 2 left middle panel) At a higher N_{min} of 30, (figure 2 right panels) most of the spurious halos

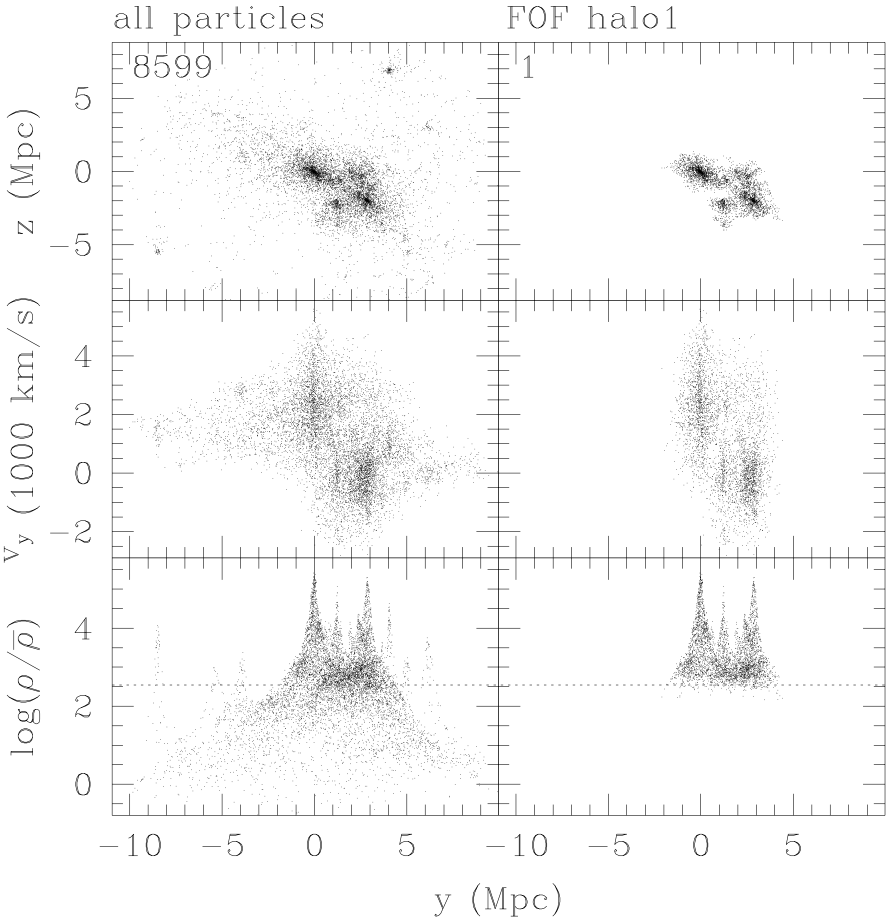

Figure 1. Particles in Model 3, and the central halo found by FOF in this model. These panels illustrate the difficulty of graphically representing data that samples a six-dimensional space (three position and three velocity components). In each panel the horizontal and vertical axes each represent one degree of freedom, and each particle contributes a single dot. In all panels, the horizontal axis is an arbitrary spatial coordinate (in this case, the y-coordinate). The panels on the left show all of the particles in the model; those on the right show the particles in the single most massive halo found by FOF. The upper panels show particle positions, projected into the spatial y-z plane. The middle panels show the same spatial coordinate and one velocity coordinate. The lower panels show the density (as calculated by IsoDen according to equation 2; see Section 4) and the same spatial coordinate. The lower panels emphasize the spatial variation and complex hierarchy of densities. It is clear from the lower right panel that the FOF method has failed to identify some of the substructure in the model, combining several distinct halos into a single halo. The horizontal line in the lower panels indicates the value of ρ_{min} corresponding to the h_{link} used for FOF.

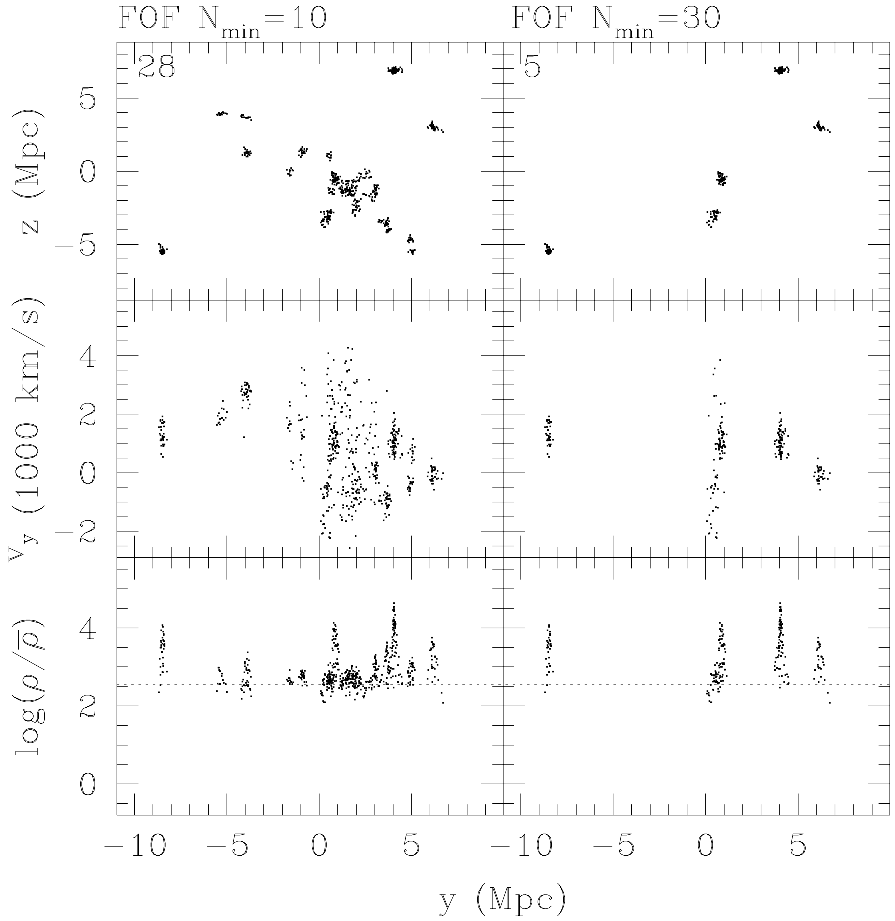

Figure 2. As Figure 1, showing halos other than the central halo, as found by FOF with N_{min} = 10 (on the left) and N_{min} = 30 (on the right). The number of halos in each case is indicated at the top left. The large spread in the velocity coordinate in the middle panels indicates that even though some groupings of particles have high spatial density, they may not be gravitationally bound, and hence do not constitute a persistent object in the simulation. Thus, the middle panels suggest that many of the selected halos on the left, and at least one of those on the right are spurious. The lower and upper panels, however, indicate likely structures on the left (identifiable with N_{min} = 10), which are rejected by the choice of N_{min} = 30 on the right. Thus, neither value of N_{min} is entirely satisfactory.

are rejected, but one remains, and in addition several real (but small) halos have been rejected.

3.1. Rejecting halos based on the minimum number of friends, $\boldsymbol{N_{fmin}}$

We suggest here a simple improvement. Replace N_{min} with a new parameter, N_{fmin}, and only accept halos that have at least one particle with at least N_{fmin} friends. The advantage of N_{fmin} over N_{min} is that diffuse, relatively low-density linked groups are more likely to be rejected, while small but compact clumps may still be accepted as real halos. The effects of N_{fmin} in Model 3 are shown in the left hand panels of Figure 3, using $N_{fmin} = 10$. In this case all of the spurious halos

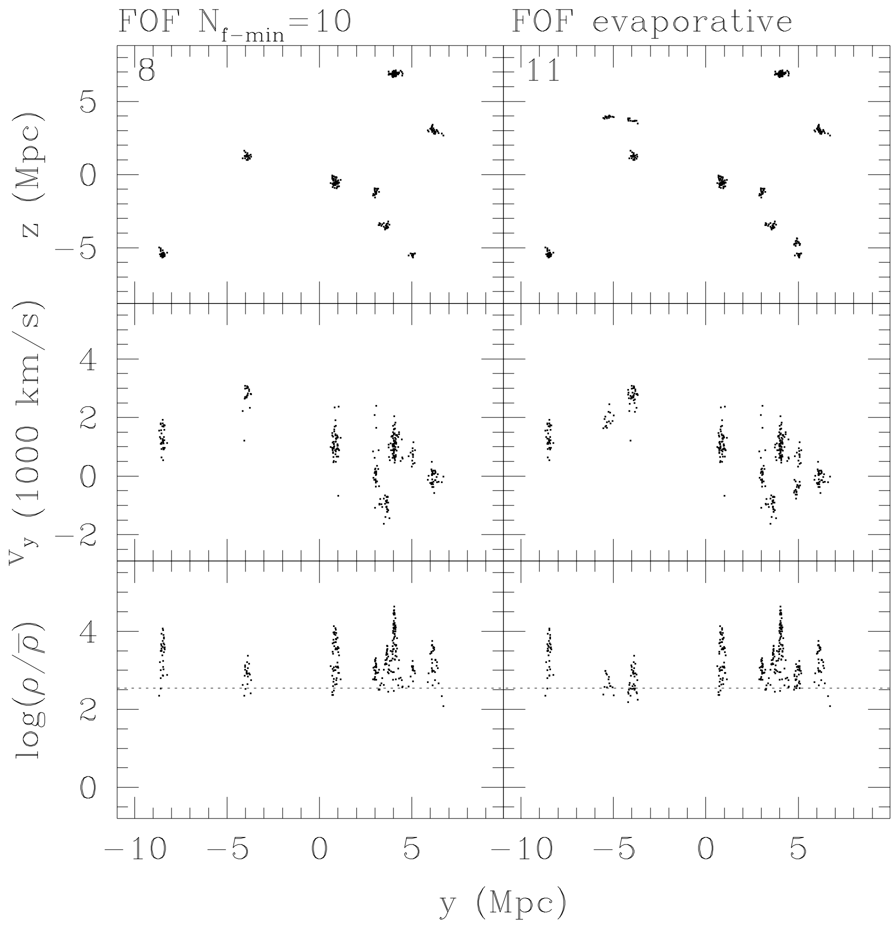

Figure 3. As Figure 2, but using the two alternative methods for rejecting spurious halos as discussed in the text. On the left are results for $N_{fmin} = 10$, and on the right are results using the evaporative method. The evaporative method finds 11 halos, all of which are guaranteed (by construction) to be gravitationally bound. In contrast the method with $N_{fmin} = 10$ requires much less computation, but finds only 8 halos (not as good as the evaporative method, but better than the conventional N_{min} method, see Figure 2).

are rejected, and more real halos are accepted than for $N_{min} = 30$.

3.2. Rejection based on binding energy

An alternative approach to rejecting spurious halos is to use the actual particle velocities to directly calculate whether putative halos are gravitationally bound. We do this using an "evaporative" method which is almost the same as that used in the DENMAX algorithm [3, 6]. In this method each putative halo is considered in isolation, and the total energy of each particle is calculated: its gravitational potential energy in the field of the other particles, plus its kinetic energy relative to the center of mass of the halo. Then the particle with the highest energy is considered. If the particle's total energy is negative then so are all particles with lower energy, so the ensemble is guaranteed to be gravitationally bound, and the halo is accepted. Otherwise the particle's energy is positive — meaning that it is on an orbit that will eventually remove it from the neighborhood of the halo. The particle is removed from the halo ("evaporated") and the process is repeated, recalculating the energies of the remaining particles to account for the loss of the removed particle. The process continues until either the halo is accepted, or the number of particles remaining drops below some minimum, in which case the halo is rejected. (In the results included in this paper the minimum number of particles is set to 10.)

The evaporative method can be considered more correct that the simple N_{min} and N_{fmin} methods (in terms of directly addressing our definition of a halo), but it does involve significantly more computation. For each halo containing N_h bodies, evaporation requires $O(N_h^2)$ operations. To save time, we assume that halos with very many particles (say $N_h > 1000$) are bound, since this is almost certainly the case. Results using the evaporative method in Model 3 are shown in the right hand panels of Figure 3. The method is very good at discriminating between genuine and spurious halos, and finds eleven genuine halos compared with only eight found by $N_{fmin} = 10$.

4. The IsoDen Method

4.1. Basic IsoDen Method

The main aim of the IsoDen method is to improve on FOF by identifying halos over a wide range of densities, thereby exploiting the full dynamic range available in the data. The motivation is similar to that of the DENMAX algorithm [3, 6], but the procedure is substantially different. Another related method is presented in [15], but it is restricted to spherical isodensity surfaces. In some respects IsoDen corresponds to applying the FOF method at a range of densities or linking lengths, a course suggested by [5], but in an integrated and consistent way.

The idea of IsoDen is to actually calculate a spatial density field defined by the particles, and then identify halo centers as local peaks in this field. Isodensity surfaces are grown around each center, to find those particles belonging to each peak. When the isodensity surfaces of different centers touch, a new composite halo and isodensity surface is created that encompasses the two constituent peaks. In order for the method to work, the density field derived from the particle distribution must be reasonably smooth, so that the interpretation of the method in terms of isodensity surfaces is valid.

The density at each particle, ρ_i, (i ranging from 1 to N), is calculated as the sum of the masses, m_j, of the nearest N_{kern} particles, divided by the volume of the sphere enclosing those particles. This is simply k-nearest neighbor density estimation with $k = N_{kern}$ [11].

$$\rho_i = \frac{1}{\frac{4}{3}\pi h^3_{i,N_{kern}}} \sum_{j \in \{N_{kern}(i)\}} m_j, \tag{2}$$

where $h_{i,N_{kern}}$ is the distance from particle i to its N^{th}_{kern} nearest neighbor and the sum is over the N_{kern} nearest neighbors of particle i. The variable smoothing scale implicit in nearest neighbor estimation is important because of the large range in densities present in the simulations. By using the nearest neighbor method with, e.g., $N_{kern} = 12$, the local resolution in the density is tailored to the actual resolution available.

In principle the IsoDen method could use alternative density measures which satisfy the requirement of spatial continuity. One possibility is the phase space density: the mass per spatial volume element per velocity volume element. This may be advantageous for cosmology simulations, because low density halos generally have small internal velocities, (e.g., see Figures 1 to 4) and hence have similar phase space densities as halos with higher spatial density. Other density estimation techniques are also viable, c.f. [10], and may be more appropriate in other problem domains. We have experimented with the generalized nearest-neighbor technique [11] but find that for the same level of uncertainty in the density field, it is more computationally expensive.

The isodensity surfaces are defined implicitly by a graph-connectivity criterion similar to that used in FOF. Each particle is implicitly linked to N_{link} nearest neighbors, where N_{link} is a parameter of the method. An isodensity surface is defined by taking all the particles above some fixed density that are also linked together, as in FOF. The value of N_{link} should be large enough that all particles are linked together when all particles are considered, i.e., the isodensity surface at zero density should comprise the entire system. N_{link} should not be unnecessarily large, since this would compromise the spatial resolution of the method. In practice values of 12 to 24 have been successfully used. Note that isodensity surfaces are not directly calculated in the method (which is explained algorithmically below), but rather are a useful concept for understanding the method.

Once densities and linking lengths have been calculated, the particles are sorted by density, and then considered in order from highest density to lowest. Our goal is to assign each particle to a halo, which are numbered sequentially from zero. The halo number for each particle is calculated based on the halo numbers of the higher density particles to which it is linked, as follows:

- If the particle under consideration is not linked to any other particles with higher density, then a new halo is created and the particle is assigned to it.
- If the higher-density linked particles all belong to the same halo, then the new particle is assigned to that halo as well.
- Otherwise, the linked particles with higher density belong to two or more distinct halos. In this case the pre-existing halos *overlap* at the density of the particle being considered. A new composite halo is generated to represent the overlap of those halos. The membership lists of the overlapping halos are recorded, and then all the particles in the overlapping halos, along with the particle under consideration, are reassigned to the new composite halo.

This procedure results in a tree of halos, where the leaves correspond to the central regions of distinct halos, and the internal nodes correspond to composite halos, defined by isodensity surfaces, where various halos overlap.

4.2. Noise Suppression in IsoDen Method

In any local spatial region of the simulation there will be statistical fluctuations in the number of particles. These fluctuations lead to errors in the calculated density field (relative to the hypothetical underlying mass distribution being sampled by the N-body particles). Such errors will result in "noise peaks", and so as with FOF, IsoDen requires a method for rejecting spurious halos. The evaporative method discussed for FOF is effective for this purpose, but requires considerable computation. As an alternative, we describe a simple statistical method which is unique to IsoDen and quite effective.

The statistical method requires that one be able to calculate the statistical uncertainty of the density estimate at each particle. For the kernel density estimation described above, the uncertainty can be estimated by assuming that the underlying density distribution is roughly uniform on scales that contain N_{kern} particles, and that the particle positions are sampled at random from this density field. Then the uncertainty in the density is just due to Poisson noise, and the statistical uncertainty, σ_i, is simply $\rho_i/\sqrt{N_{kern}}$. That is, the relative uncertainty, σ_i/ρ_i, is a constant, $1/\sqrt{N_{kern}}$. This is an important feature of nearest neighbor density estimation: in high density regions the spatial resolution is improved (i.e. smaller h_i) while maintaining constant relative uncertainty in the density estimates.

When each new halo is created, the halo is designated *tentative*, and the particle that creates it defines the halo's central (peak) density, ρ_c. Tentative halos will become either *genuine* halos, or will be eliminated based on a statistical criterion. When a tentative halo overlaps with another halo we apply a somewhat *ad hoc* criterion akin to a statistical significance test. We compare ρ_c, with ρ_o, the density at which the overlap is detected, plus $n\sigma_o$, the statistical uncertainty at the overlap density: i.e., if

$$\rho_c > \rho_o + n\sigma_o \tag{3}$$

we accept the peak as genuine. Otherwise it is rejected. Since the probability distribution of ρ_c is somewhat difficult to define, we cannot precisely define the significance of this test. Empirically, we find that "three sigma" peaks, i.e., $n = 3$ are almost always genuine in the sense that they pass the physically motivated evaporative test.

If a tentative halo passes this test, it becomes genuine and is recorded as an independent object (a leaf of the halo-tree) which is contained within the larger composite object that is created by the overlap. If it fails the test, the tentative halo is rejected. In either case, all particles in the overlapping halos are renumbered with the new composite halo-number. A composite halo is genuine if and only if any of its component halos are genuine.

4.3. Limitations of IsoDen

One of the limitations of IsoDen is that there are several parameters whose value is somewhat arbitrary: N_{kern}, N_{link}, and the "significance" parameter n.

The parameter N_{kern} is particularly important since it determines how much smoothing is done to the particle density field. In particular, IsoDen is unlikely to detect halos which have fewer particles than N_{kern}. Also, the peak densities and overlap densities, which are used in the noise suppression method, will be affected by changing N_{kern}. Notice that N_{kern} only enters the density estimation step. An alternative density estimator might have different parameters, but generally, some free parameter(s) in the density estimator will govern the minimum size of peaks that are discovered in the density field.

The parameter N_{link} has an effect similar to N_{kern}. If, N_{link} is too large, then a bona fide peak whose central particle is an N_{link}-neighbor of a higher density particle will be overlooked by the method. If N_{link} is too small, then overlaps are not detected at all, and halos are detected in isolation rather than as part of a hierarchy. Our limited experience is that the results are not strongly sensitive to N_{link} in the range 12 to 24.

The "significance" parameter, n, represents a tradeoff between failing to detect small halos and wrongly "detecting" spurious halos. Since the test is not actually a rigorous statistical test, an appropriate value of n needs to be empirically determined, and the best value may depend in a subtle way on the other parameters. It is even possible that the best value could vary for different simulations or for different spatial or temporal regions of a single simulation. One option could be to use the evaporative method, which is motivated by the discipline-specific knowledge about the problem, to determine an appropriate value of n in various circumstances.

Both methods for noise suppression make some assumptions about the nature of the particle position and velocity distributions in N-body simulations — e.g. specific sorts of randomness. While these assumptions are based on experience with the simulation data, and seem reasonable, it is possible that they could be violated under some conditions.

The method used by IsoDen to estimate densities has been chosen from a practical point of view as an effective way to deal with resolution in regions of vastly different density. However there are alternative methods available for density estimation which we have not examined, and would could ultimately prove even more effective.

4.4. Test Results of IsoDen Method

Figure 4 shows results obtained using IsoDen for Model 3, using both statistical and evaporative methods to reject spurious halos. In this particular case *all* of the halos found by IsoDen turn out to be real, in terms of their internal velocities (when examined individually). This is the case for both methods of noise suppression. Also, IsoDen distinguishes halos even in the high density central region of the cluster — the single large central halo found by FOF is split into 9 smaller halos by IsoDen (or 10 halos using the evaporative test). All halos found by FOF using N_{fmin} or evaporative testing are also found by IsoDen. The only difference between the IsoDen results using the two noise suppression tests is that the evaporative method identifies three extra halos which did not pass the significance test with $n = 3$.

5. Parallel Implementation

As presented in Section 4, the IsoDen method sorts particles in order of decreasing density, and then examines them in that order to generate a list of halos. At first glance, this procedure is inherently sequential, since changing the order of operations would significantly alter the result. Thus, we are forced to create an alternative formulation that achieves exactly the same result, but that allows for a mostly parallel implementation.

The initial step is the same. We calculate densities and linking lengths for individual particles. This calculation uses the same techniques, and is implemented using the same libraries, as the the density calculation required by the Smooth Particle Hydrodynamics method [13]. By using the parallel tree libraries, it parallelizes immediately. The library handles all data structure manipulation and explicit communication on message passing parallel architectures. The library was originally designed to implement parallel "Fast" summation algorithms, e.g. [1], for the computation of gravitational interactions in $O(N \log N)$ time, but the data structures are far more general. The library has been ported to a wide variety of systems including message-passing and shared-memory machines, as well as networks of workstations and uniprocessor systems. In particular, the results presented here for FOF and IsoDen methods were obtained with library implementations which managed all parallelism and data distribution on single and multi-processor SPARC workstations, a 32-node CM-5 at the Australian National University and a 512-node Intel Paragon at Caltech.

A crucial feature of the density-estimation problem is that the particle positions and masses remain fixed throughout the algorithm, and thus may be freely copied between processors without concern for coherence. In fact, the library addresses a slightly more general case which occurs in dynamical simulations — particle positions may updated only after an implicit or explicit barrier synchronization. That

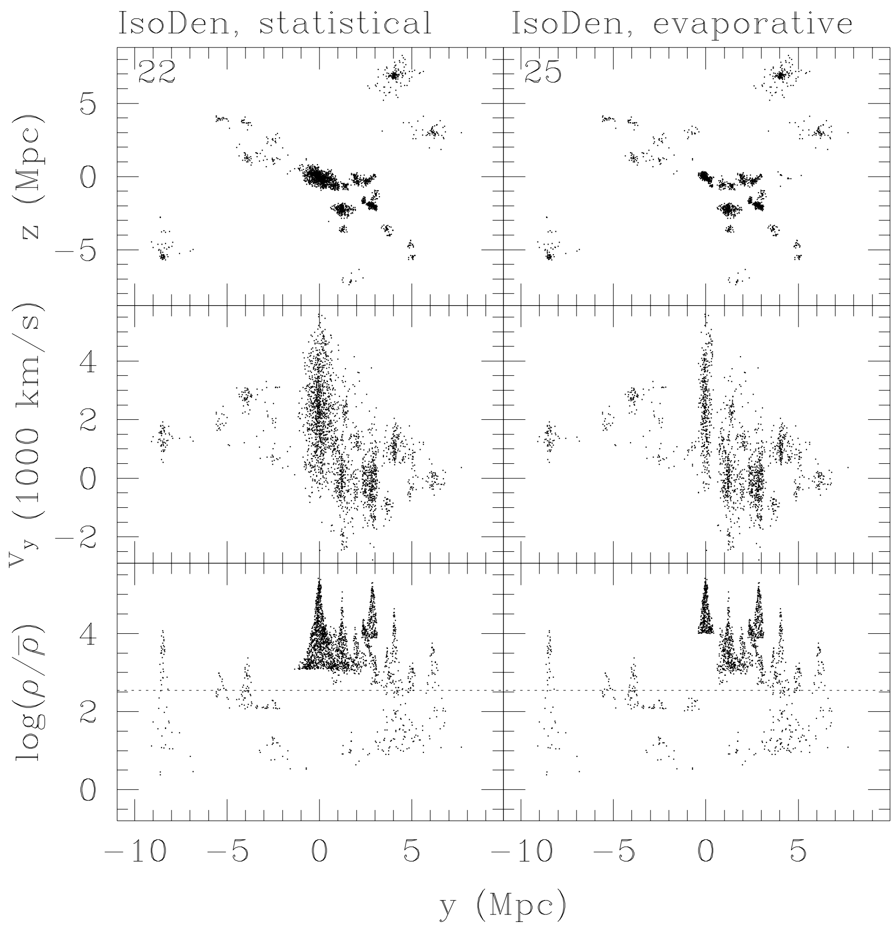

Figure 4. Results of the IsoDen method for Model 3, using the same format as Figure 1. The panels on the left show results using the statistical noise suppression test; those on the right use the evaporative test. For each halo only those particles with densities between ρ_c, the halo peak (central) density, and ρ_o, the density of the first overlap of that halo, are shown.

is, forces (or densities) are computed under the assumption that all data is static. A barrier is reached, after which the positions may be altered by the dynamics, followed by another barrier indicating that all positions have been updated and another force calculation, etc.

The parallel tree library distributes the data so that a particular result, e.g., a density estimate for a particular particle is computed by only one processor. The assignment of particles to processors must satisfy two competing goals: the processing load must be balanced, and the necessary interprocessor communication must be minimized. Load balancing of highly irregular data sets is achieved by sorting particle positions along a Peano-Hilbert curve (see Figure 7a), and then chopping the curve into $N_{processor}$ equal pieces. This has the effect of achieving load balance while maintaining spatial locality (and thereby minimizing communication overhead).

Neighbor-finding (and hence evaluation of equation 2), is done by constructing an adaptive oct-tree with individual particles at the leaves and internal nodes corresponding to cubical regions of space. The tree is traversed once for each particle in time proportional to $\log_8 N$, so that all neighbors are found in $N \log N$ time. In parallel, the tree is constructed with "remote" pointers to cells that are stored on other processors. When a particular traversal encounters such a pointer, a communication is initiated requesting the data from the remote processor. The substantial latency of such requests is overcome by a) bundling requests together into larger blocks before actually beginning interprocessor communication and b) working on another branch of the tree while the request is being processed. Furthermore, once requested, the contents of a "remote" pointer are stored locally, so that subsequent visits to the same cell do not incur the cost of interprocessor communication. This explains the value of spatial locality in the assignment of particles to processors. The neighbor-sets of nearby particles have a great deal of overlap, so that the communication cost of obtaining the neighbors from remote processors is amortized over all the local particles that are neighbors of the remote ones. All of the rather substantial bookkeeping involved is handled by the parallel tree library.

Upon completion of the density estimation phase, every particle has a list of N_{link} other particles to which it is linked. Storing these lists for all particles at once would significantly increase memory requirements. Therefore, we traverse the tree multiple times rather than exploiting the "obvious" optimization of recording explicit lists of linked particles.

For each particle, we identify the *highest density neighbor* (HDN), i.e., highest density member of the set of particles to which the given particle is directly linked. Some particles are their own HDN. These are precisely the *central* particles which define ρ_c for the tentative halos in the sequential formulation, and we give them a unique halo number.

Now we scan the list of particles until we have assigned a *preliminary halo number* to each particle. For each particle, we check its HDN. If its HDN has a preliminary halo number assigned, then the particle inherits that preliminary halo number. If the particle's HDN is not yet assigned, we revisit the particle again on the next iteration. Note that the graph defined by the neighbor relationship is highly interconnected (every particle is connected to N_{link} others), so only a few iterations are required to propagate the preliminary halo numbers to the entire system. Some interprocessor communication is required at the start of each iteration to propagate new information about particles that have been assigned halo numbers on the previous iteration.

Next, we examine the set of particles and links again, and for each pair of preliminary halo numbers, we identify the highest-density particle that is assigned to one of the two halos, and is linked to a higher density particle that is assigned to the other halo. The particles identified in this way are the overlap particles from the sequential formulation. Note that most pairs of halos are not linked at all, so this step only requires $O(N_{halo})$ space. It is easily accomplished by visiting all of the particle-particle links, and maintaining a sparse data structure that records the highest-density particle so far encountered that links each pair of halos. In parallel, when all processors are done with their own data, these local data structures are combined to produce a global structure.

Now it is possible to construct the tree of halos, where the leaves correspond to central regions of isolated halos and the internal nodes correspond to composite halos that meet at overlap particles. One sorts the overlap particles in order of decreasing density, and defines a new composite halo for each overlap. As composite halos are created, the overlap points merge, so that if A and B are merged into M, and they both overlapped with C, then M also overlaps with C, at the maximum of the two previous overlaps between A-C, and B-C. Noise suppression criteria can be implemented here as well, i.e., preliminary halos may be regarded as tentative until their density exceeds a statistical threshold over the background density, or until they pass the evaporative test. Construction of the tree of halos is, unfortunately, inherently sequential, because the overlaps must be treated in order of decreasing density. It is far faster than a naive implementation of the formulation in Section 4, though, because we have identified the overlap particles in parallel, and it is only the tree construction that is sequential. Furthermore, the tree construction only uses data related to the overlaps. The much larger particle data set can be ignored while the logical structure of the tree of halos is constructed from the maximum density overlaps. In practice, it is acceptably fast, even on highly parallel systems. See Section 6 for details of the scalability.

Once the tree of halos has been constructed, with density levels indicating where halos merge into one another, it is a simple matter to distribute the tree to the processors in a parallel system, and then assign each particle to a final, genuine halo in parallel. Starting with its preliminary halo number (at a leaf of the tree), one simply ascends the tree of halos (toward the root) until the overlap density is less than the particle's own density. The particle is assigned to the composite halo defined by this location in the tree.

Friends of friends can be implemented almost as a special case of IsoDen, using constant kernel and linking lengths, and using the number of friends as a measure of density. In the FOF case all halo overlaps result in the overlapping halos being merged together, and the substructure before the overlap is forgotten.

6. Performance Results

Figure 5 shows the halos found by IsoDen in Model 1. The halo finder ran for approximately 20 minutes on a 512 node Paragon at Caltech, and required over 5 gigabytes of memory. This computation would have been prohibitively time-consuming on a uniprocessor system — assuming we could have found one with sufficient memory!

Figure 6 shows some additional timing results from halo finding on a Paragon. Most of the calculations are shown in the top panel, and show good scaling, which

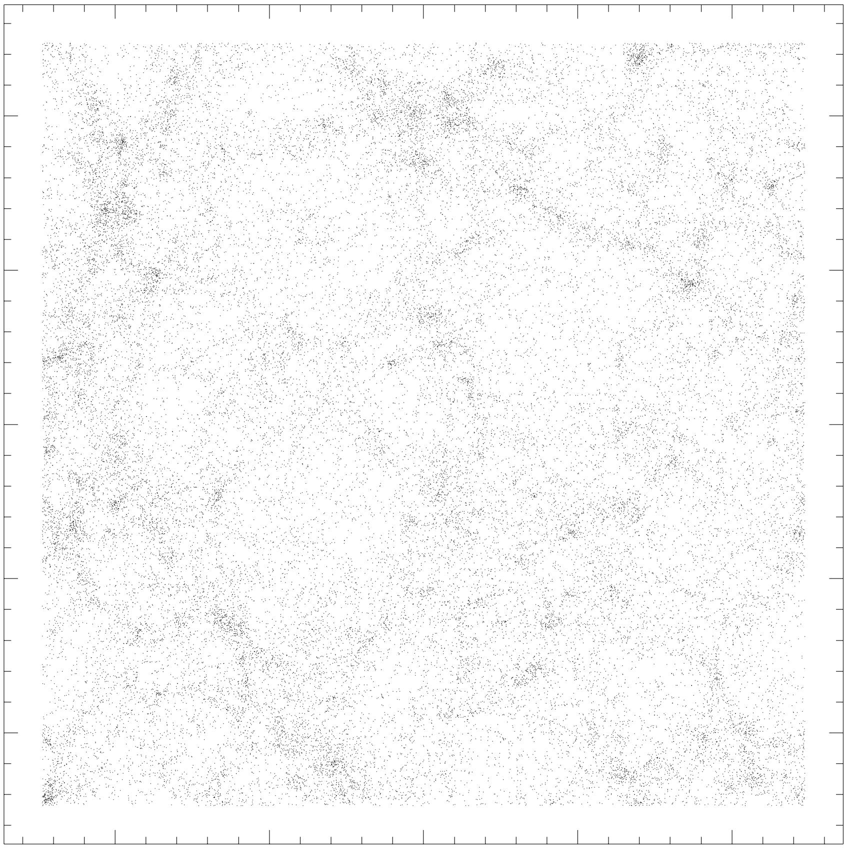

Figure 5. The positions of halo centers as found by IsoDen from Model 1, projected into an arbitrary plane; the box is 250 Mpc on a side. Only halos with central density at least 178 times the background density of the system are shown, since lower density halos are poorly resolved for astrophysical purposes. There are 43,727 halos shown, and a further 16,631 low density halos were identified.

is expected since for these calculations most of the parallelization is done effectively via the tree library. The last few steps of the method, which are more difficult to implement in parallel, are shown separately in the lower panel. Even for these steps the scaling is reasonable.

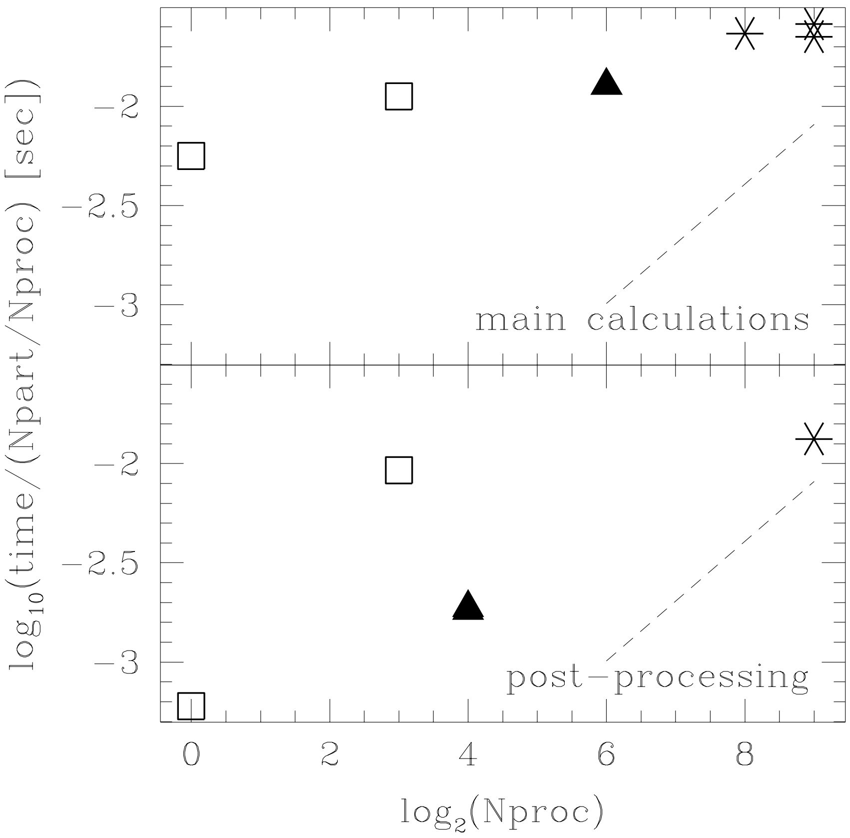

Figure 6. Timing results for the IsoDen method. The *time* is the average CPU time on each processor, *Npart* is the total number of particles, and *Nproc* the number of processors. The different point shapes indicate different models (in particular, different Npart), from table 1: stars are model 1, triangles are model 2, and squares are model 3. The dashed line indicates the slope for constant time independent of Nproc. The top panel shows the time for all calculations up to and including the calculation of the overlaps of preliminary halos (see Section 5). The lower panel shows the time for the remaining steps: the construction of the tree of halos, the application of noise suppression tests, and the calculation of genuine halo numbers.

7. Future Directions

7.1. Out-of-Core Computing

The development of out-of-core techniques suitable for processing highly irregular, unstructured data, such as are generated by cosmological N-body simulations, is crucial to manipulating a dataset of sufficient size. We have recently obtained extremely promising results applying parallel, out-of-core techniques to the irregular and dynamic treecode methods that are used within the N-body integrator itself [9]. Essentially, the problem reduces to one of guaranteeing spatial and temporal locality so that access to secondary storage is necessary only rarely, and once loaded from secondary storage, a given datum is likely to be re-used many times in primary storage before being flushed. The solution is exactly the same one that has been employed to good effect to achieve load balance and tolerate latency in parallel implementations of treecodes — organize and divide the data along a space-filling curve that approximately preserves spatial locality even if the underlying data is highly clustered and irregular. The crucial observation is that by placing the pages dynamically on disk in an order determined by a continuous space-filling curve, e.g., a Peano-Hilbert curve, (see Figure 7) and ordering our access to data along the same curve, we preserve the spatial and temporal locality in the algorithm. Since the space-filling curve is continuous, objects that are near in space, and hence likely to be linked together by the algorithms, are also near in primary or secondary storage. By visiting objects in order along the space-filling curve, objects that were recently brought into primary storage, either because they were directly addressed, or because they are on the same page as a recently addressed object, are likely to be reused. Since the parallel implementation of the spatial halo finder uses the same data and control structures as the N-body code to tolerate irregularity and achieve parallelism, it can be adapted to use out-of-core storage in the same way that has worked for the N-body code itself.

7.2. Adaptive Temporal Method for Halo Tracking

While the techniques described above provide high accuracy halo extraction tools, they are limited to single-time, 3-dimensional slices through the overall data set. We estimated that extracting halos from particle positions by this method will require tens of minutes on the cluster of 16 PCs. This is not substantial, but when used across a data set of a few thousand frames, the time to completion becomes prohibitive (several months). Therefore an adaptive time domain algorithm is conceived and is being developed that augments the pure spatial method.

From an initial halo map derived from the spatial method, succeeding frames are used only to adaptively track identified objects. A centroid for each halo is recomputed for each frame and the population of particles is updated. This technique is very fast and requires less than a minute per frame to compute. After on the order of a hundred frames, a full spatial particle reduction is performed to rederive halos at that time step. These are correlated with the projected objects from the temporal method. New objects are identified as a result and a partial temporal roll-back sequence is then performed (going backward in simulated time) for those new halos, thus tracking them to their genesis.

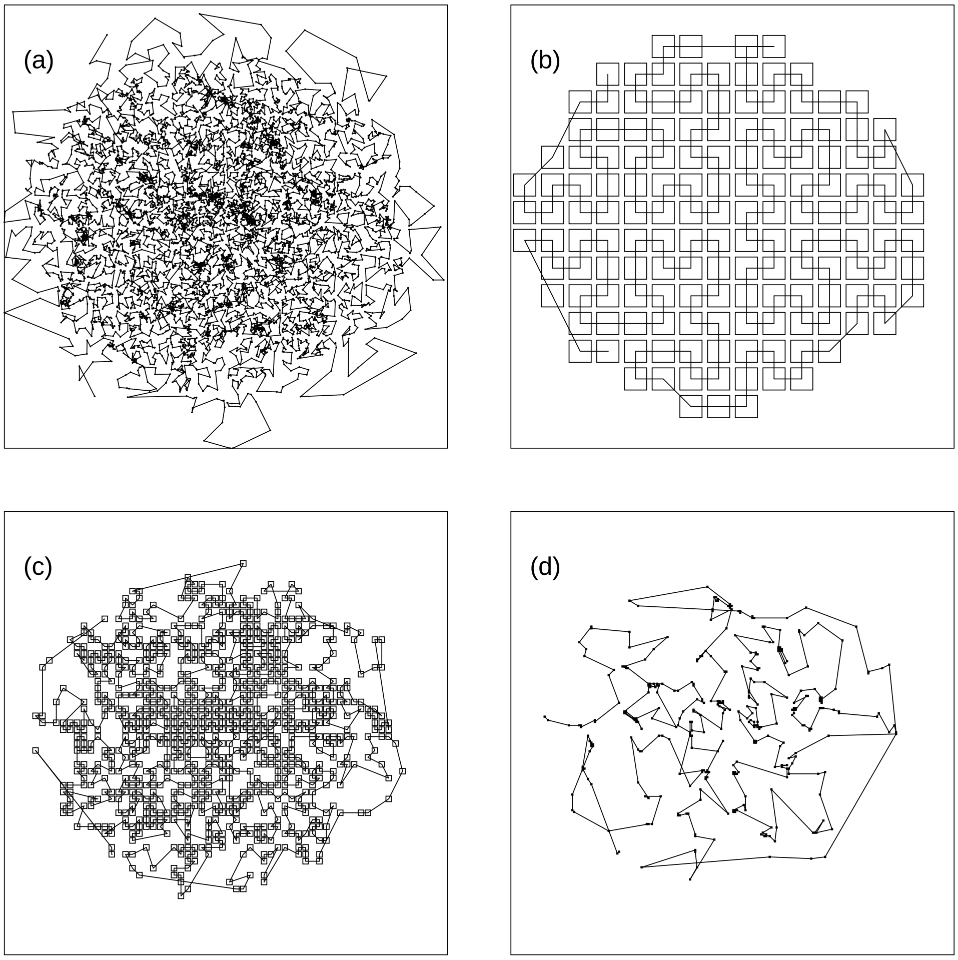

Figure 7. A path through the positions of 10000 highly clustered particles in two dimensions is shown in (a). The path is obtained by locating each particle along a space-filling Peano-Hilbert curve. Levels 4, 6 and 8 of an adaptive quad-tree are shown in (b), (c), and (d). The squares represent "cells" in an adaptive quad-tree that is used to compute densities and locate neighbors in $O(N \log N)$ time. The whole tree is obtained by "stacking" these levels, along with those not shown. Again, the path connecting the cells is derived from a Peano-Hilbert curve. In this case, the curve indicates the order in which data are stored in both primary and secondary memory. Each level of the tree is stored separately, so that data within the same level is always contiguous. When particles are visited in the order implied by (a), they cause pages to be moved from secondary storage which hold data that is contiguous along the curves represented in (b), (c), (d). The orderings guarantee that the data is reused many times before being discarded, allowing the algorithm to tolerate the modest bandwidth and extreme latencies characteristic of out-of-core calculation. (Two dimensions are used purely for ease of graphical representation. The implementation is fully three-dimensional.)

8. Conclusions

We have presented a new IsoDen algorithm for finding halos in N-body cosmology simulations, and described an implementation on parallel computers. This new method has advantages compared to the friends of friends (FOF) algorithm, which has also been implemented in parallel. In particular the IsoDen method robustly finds density peaks even in the high density central regions of clusters. Our tests indicate that these halos are "real" in the sense of being gravitationally bound, persistent objects in the simulation, so the IsoDen method is a genuine knowledge discovery process. The use of a statistical estimate of the uncertainty in density estimation to distinguish real peaks from chance associations is novel and effective, even though it lacks a firm theoretical foundation.

By implementing these methods on parallel machines we are able to use them to begin the analysis of the massive datasets produced by modern high resolution N-body cosmology simulations. This will allow us to address the task of accurately interpreting these simulations, to understand the physical processes involved in the formation and evolution of dark matter halos, and to compare the simulations to astronomical observations.

We also consider issues involved in implementing these techniques on inexpensive clusters of commodity processors, using out-of-core techniques to eliminate the need for impractically large amounts of memory. Our implementation of the halo finding techniques in terms of a parallel tree library originally designed for N-body simulation suggests that out-of-core techniques will be practical and viable. A tree library that uses out-of-core techniques is under development, and will form the basis of a parallel, out-of-core implementation of the FOF and IsoDen halo finders. Tracking halos in time, and correlating the results using an *ab initio* halo finder like IsoDen only at widely separated times will further improve performance.

9. Acknowledgments

The Halo World project is supported by NASA grant NAG5-4203. Part of the research reported here was performed using the Intel Paragon system operated by the Center for Advanced Computing Research at Caltech; access to this facility was provided by Caltech. We also acknowledge computing resources provided by the Australian National University Supercomputer Facility. DWP acknowledges support from an Australian Postgraduate Award, a Caltech Graduate Research Assistantship, and a European Southern Observatory Studentship.

References

1. Joshua E. Barnes and Piet Hut. A hierarchical O(NlogN) force-calculation algorithm. *Nature*, 324:446–449, 1986.
2. Donald J. Becker, Thomas Sterling, Daniel Savarese, John E. Dorband, Udaya A. Ranawake, and Charles V. Packer. Beowulf: a parallel workstation for scientific computation. In Tse yun Feng, editor, *Proceedings of the 1995 International Conference on Parallel Processing, Volume 1: Architecture*. CRC Press LLC, 1995.

3. E. Bertschinger and J. M. Gelb. Cosmological N-body simulations. *Computers in Physics*, 5:164–179, 1991.
4. R. G. Carlberg. Velocity bias in clusters. *Astrophysical Journal*, 433:468–478, 1994.
5. M. Davis, G. Efstathiou, C. S. Frenk, and S. D. M. White. The evolution of large-scale structure in a universe dominated by cold dark matter. *Astrophysical Journal*, 292:371–394, 1985.
6. J. M. Gelb and E. Bertschinger. Cold dark matter 1: The formation of dark halos. *Astrophysical Journal*, 436:467–490, 1994.
7. P. J. E. Peebles. *Principles of Physical Cosmology*. Princeton, 1993.
8. David W. Pfitzner. Density profiles of dark matter halos formed in dissipationless cosmology simulations. In M. Arnaboldi, G. S. Da Costa, and P. Saha, editors, *The Second Stromlo Symposium: The Nature of Elliptical Galaxies*, pages 109–110, San Francisco, 1996. Astronomical Society of the Pacific.
9. John K. Salmon and Michael S. Warren. Parallel out-of-core methods for N-body simulation. In Michael Heath, Virginia Torczon, et al., editors, *Eigth SIAM Conference on Parallel Processing for Scientific Computing*. SIAM, 1997.
10. David W. Scott. *Multivariate density estimation: theory, practice and visualization*. Wiley-Interscience, 1992.
11. B. W. Silverman. *Density estimation for statistics and data analysis*. Chapman and Hall, 1986.
12. F. J. Summers, M. Davis, and A. E. Evrard. Galaxy tracers and velocity bias. *Astrophysical Journal*, 454:1–14, 1995.
13. Michael S. Warren and John K. Salmon. A portable, parallel particle program. *Computer Physics Communications*, 87:266–290, 1995.
14. Michael S. Warren, John K. Salmon, Donald J. Becker, M. Patrick Goda, Thomas Sterling, and Grégoire S. Winckelmans. Pentiumpro inside: I. a treecode at 430 gflops on ASCI red, II. price/performance of $50/Mflop on Loki and Hyglac. In *Supercomputing '97*, Los Alamitos, 1997. IEEE Comp. Soc. in press.
15. W. H. Zurek, P. J. Quinn, J. K. Salmon, and M. S. Warren. Large scale structure after COBE: Peculiar velocities and correlations of cold dark matter halos. *Astrophysical Journal*, 431:559–568, 1994.

Contributing Authors

David W. Pfitzner Obtained degree of Bachelor of Science with Honours, at The University of Adelaide, 1994. Currently studying for degree of Doctor of Philosophy in Astronomy and Astrophysics, at Mount Stromlo and Siding Spring Observatories, The Australian National University.

John K. Salmon received his Ph.D. from Caltech in 1991. His research is in parallel scientific computation, and the use of massively parallel supercomputers for applications in astrophysics and fluid dynamics. In 1992 he shared the Gordon Bell prize for achievement in large scale computing. Recently, he has been studying out-of-core techniques and the use of commodity processors to achieve cost-effective large-scale computing.

Thomas Sterling has been engaged in research related to parallel computer architecture, system software, and evaluation for more than a decade. He was a key contributor to the design, implementation, and testing of several experimental parallel architectures. His current research focuses

, , 23?? ()

on innovative approaches to achieving peta-scale (i.e., 10^{15} operations per second and/or 10^{15} bytes of storage) computing. The Hybrid Technology Multi Threaded (HTMT) architecture combines advanced technologies from superconducting logic, optical switching, processor-in-memory fabrication, multithreaded programming and optical holographic storage into a single system. On a more mundane level, he has also pioneered the Beowulf approach to commodity parallel processing. Since completing his Ph.D. as a Hertz Fellow from MIT in 1984, Dr. Sterling has done research at the Harris Corporation's Advanced Technology Department, the IDA Supercomputing Research Center and the USRA Center for Excellence in Space Data and Information Sciences at the Goddard Space Flight Center. He now holds a joint research appointment at the California Institute of Technology and the NASA Jet Propulsion. He holds six patents, is the co-author of two books and has published dozens of papers in the field of parallel computing.