

Using N -body Algorithms for Interference Computation in Wireless Cellular Simulations *

L. Felipe Perrone
Dept. of Computer Science
The College of William and Mary
perrone@cs.wm.edu

David M. Nicol
Dept. of Computer Science
Dartmouth College
nicol@cs.dartmouth.edu

Abstract

A comprehensive simulation model of wireless cellular networks must include the computation of transmitter power levels. In such systems, as time evolves, powers are continuously updated to minimize interference and maintain signal quality. Transmitters operate at the minimum power required to meet a target signal to noise ratio (SNR), which, in the real system, can be promptly estimated since the values involved come from direct measurements. In a simulation model, however, the interference over each receiver is a quantity that must be computed and the associated costs are not low. A system with N pairs of transmitters and receivers requires that $O(N^2)$ pairwise interactions be computed; it's easy to see how very large the workload is when we consider that, in order to advance simulated time by one second, this large computation may have to be performed hundreds of times. In this paper, we show that techniques devised for the simulation of systems of self-gravitating bodies (N -body problem) can be successfully applied to reduce the complexity of interference computations in simulations of wireless systems. However, our experiments suggest simple distance-based truncation may be the superior method.

1. Introduction

As wireless cellular systems become increasingly more popular, more research effort is focused on their design and the need for ever faster and more accurate simulators becomes evident. In the last 5 years, the simulation community has witnessed a trend for the development of larger and more comprehensive models for wireless networks.

Several research groups throughout the world have been hard at work developing simulators that focus not only on

one or another component of a wireless network, but on the problem as a whole. The WINLAB group at Rutgers University [10, 11, 15, 16] has developed WiPPET, a general parallel simulation testbed that allows users to easily put together complex models using building blocks designed by experts in each different sub-area. Drawing from a database of components, the simulationist can then concentrate on high-level model design rather than dealing with the intricacies of programming the simulator's core. Another group at UCLA has been involved in a project of similar scope called GloMoSim [3, 18]. While WiPPET relies on the optimistic synchronization protocol of Georgia Tech Time Warp (GTW) [6] for parallel execution, GloMoSim works at the other end of the spectrum utilizing conservative algorithms.

The significance of these works has been paramount to making the construction of reliable simulation models of wireless systems more accessible. However, there are still several performance issues at the simulator core level that bear the need for deep investigation. Until these problems have been properly addressed and solved, the simulation of wireless models of realistic scale will remain a computational burden of almost intractable proportions.

On this other front, the contributions by the group at the Royal Institute of Technology, in Sweden, have been key. First, they have shown the efficiency of partitioning wireless models by channel rather than by geography for parallel execution [12, 14]. Later on, they launched an investigation on what can be regarded as the bottleneck of a comprehensive wireless simulation: interference computation. In their paper, Liljenstam and Ayani [13] consider a model that includes adaptive power control and simplify interference computations by a truncation of interaction radius. The idea stems from the fact that radio signals decay exponentially fast with increasing distance between transmitters and receivers. They have shown that the careful application of this concept can produce small errors, while dramatically reducing the asymptotic complexity of computations. They also report that analysis of the errors introduced by truncation must be done on a case-by-case basis.

*This research was supported in part by NSF grants ANI-98-08964, EIA-98-02068 and DARPA contract N66001-96-C-8530.

We now proceed to show how the same problem can be approached using a more sophisticated technique. There is a strong similarity in the simulation of wireless radio and astrophysical models and use a well-known fast solution for this second problem to attack the first. The gravitational interaction between two bodies is inversely proportional to a power of the distance between them. The analogy with radio signal propagation is obvious, for their strength also decays exponentially fast with the distance between transmitter and receiver. Our objective is to reduce the computational cost of interference calculation from the brute-force method derived directly from interference equations, while systematically controlling the error introduced by doing so.

In the remainder of this paper we develop this idea further and provide an experimental analysis of its validity. Section 2 presents our system model together with relevant background concepts in wireless cellular systems. Later on, in Section 3, we present the *N-body problem* in more detail, showing algorithms for its solution and outlining the implications of their use in our simulations. The results of our experiments are discussed in Section 4 and we conclude in Section 5 introducing ideas for future research.

2 Simulation Model

We define the geographical space for our simulations as a plane, where all base stations (BS) and mobiles stations (MS) are contained. The coverage area of each BS is represented by a regular hexagon and all cells have the same dimensions. Furthermore, in order to avoid the occurrence of edge-effects in this model, we arrange the cells into a torus.

Among the many possible models of user mobility, we have chosen to impose a “street” layout on the simulation space. Mobiles are allowed to traverse the space exclusively on the diagonals of the hexagons. Wherever diagonals intersect, that is, at cell centers and vertices, a mobile chooses a new direction of movement with equal likelihood. The speed of travel is constant on a straight line segment between two dots, but a new one may be chosen at every intersection point.

It is well known that electromagnetic waves are subject to an exponential decay with distance. To model propagation on the path between transmitter i and receiver j allowing for the representation of shadow fading, we describe path loss as:

$$G_{ij}(t) = \frac{L_{ij}(t) g_{ij}(t)}{d_{ij}(t)^\alpha} \quad (1)$$

where $d_{ij}(t)$ is the distance between i and j at time t , attenuation exponent $\alpha = 4$ on the path (in the absence of obstacles) and $g_{ij}(t)$ allows for modeling fine descriptions of terrain features given the current position of MS and BS.

$L_{ij}(t) \sim \text{Lognormal}(a, b)$ is a random variable which accounts for transient changes in the propagation path and follows the standard definition: $L_{ij} \sim \exp(a + bX)$ where $X \sim \text{Normal}(0, 1)$. We make it a function of time to indicate the fact that the variable is re-sampled at every instant t when path loss is evaluated.

We assume that the system works with the notion of *channels*: whether the spectrum is shared through *frequency division multiple access* (FDMA) and/or *time division multiple access* (TDMA) this becomes a straightforward abstraction. Each channel connects a MS-BS pair using two components: an *uplink*, for communication from MS to BS and a *downlink* for the reverse direction. Other important assumptions we make are that all channels are orthogonal, that is, there is no adjacent channel interference, and that there is no multipath fading.

The only sources of spurious signals that define interference, in our model, are *co-channel interference* and *thermal noise*. The first of these is defined as the summation of signals from other BS-MS pairs using the same channel measured at the receiver’s location. The second, ν , accounts for noise generated by semiconductor devices inside the receiver itself. Together these two define the total instantaneous noise for each receiver j using a channel c :

$$v_j^c(t) = \nu + \sum_{i \neq j} G_{ij}(t) p_i^c(t) \quad (2)$$

where $p_i^c(t)$ is transmitter i ’s power level at time t .

Finally, we complete the description of our simulation model by defining power control and channel allocation policies. Among the many choices, we have opted for the *local autonomous dynamic channel allocation* (LADCA) scheme by Foschini and Miljanic [7, 8]. This algorithm of simple implementation was developed for FDMA, TDMA or GSM-like systems (FDMA/TDMA hybrids) and gives us the opportunity to create the scenario for realistic simulation experiments.

LADCA works on a discretized time scale and requires that power levels be updated frequently, at short time-steps. When a transmitter’s power level reaches maximum value and the target SNR ρ still cannot be met, a drastic action such as a switch to another channel, a call block or drop is triggered to alleviate the load on the channel and allow other calls to proceed. In short, the method consists of looking at the system’s state at a discrete point in time and then increasing or decreasing power levels according to the difference between current SNR and target SNR. This is formally expressed by equation 3, where $\beta > 0$ is a proportionality factor.

$$p_i^c(k+1) = p_i^c(k) - \beta \left[p_i^c(k) - \rho \frac{v_i^c(k)}{G_{ii}(k)} \right] \quad (3)$$

To obtain the new power level for transmitter i after a time increment, first MS’s positions are updated and then the

equation is iterated until a fixed point is reached. If there exists a power level less than or equal to the maximum that allows ρ to be met, this algorithm is guaranteed to find it independently of the initial state.

3 N -body Methods

Originally, the N -body problem was defined as the simulation of a system with a large number self-gravitating bodies. Each element would be a planet, galaxy or some object with a known mass subject to the laws of gravitation. Given the mass, initial position and velocity of each body, the goal is to determine the trajectory of each one as time progresses.

The first step in the process is to compute the total gravitational force over each body, which is a summation of pairwise interactions with every other element in the system. Once force has been calculated, the acceleration vector can be obtained by Newton's second law and the trajectory of each body computed by numerical integration over time.

The force between two particles with masses m_1 and m_2 is proportional to the product $m_1 m_2$ divided by the square of the distance between them. To compute the evolution of trajectories in time, a time-stepping simulation can be designed. Starting from some value t , time is advanced by Δ units; positions at $t + \Delta$ are computed according to velocities known at time t . Once new positions are known, the force over each body must be computed so that corresponding acceleration can be updated. This scheme produces valid results only as long as the chosen Δ is small enough so that accelerations do not change much during that interval.

The similarities between interference computation in wireless simulations and force computation in N -body problems are strong and evident:

- Given a number of elements N , an $O(N^2)$ number of pairwise interactions must be calculated,
- The strength of the interaction within each pair of elements is inversely proportional to the distance that separates them raised to an exponent greater than or equal to 2 and, finally,
- Both problems are described by differential equations in time which are solvable by numerical integration. In the two cases the equations are derived from the assumption that a quantity remains constant throughout the time-step (force in the gravitational problem and noise in the wireless network).

From this analogy we see that methods to solve the N -body problem can be easily adapted to compute interference in the simulation of cellular systems. The many algorithms for N -body simulation [2, 4, 5, 9] in the literature have one

point in common: the use of a clever data structure that allows a drastic reduction in run-time. The time complexity can be reduced from $O(N^2)$ to $O(N \log N)$ or even $O(N)$, which represents a major improvement over the naïve algorithm when N is large.

The data structures, usually quad-trees for the two-dimensional case and an oct-trees for the three-dimensional case, implement a simple concept. Suppose we are computing the total interaction over a certain particle and want to consider the effect produced by a distant cluster of particles. If the distance d that separates the cluster and the particle is sufficiently large, we can produce a good approximation substituting the whole cluster by one single particle (whose mass is equal to that of the summation of individual masses) located at its center of mass. Instead of computing a number of pairwise interactions between the particle of interest and each element of the cluster, we compute only one interaction with the imaginary particle.

The summation of interactions in one point in space can be expanded into an infinite Taylor series: the first term is named monopole, the second dipole, the third quadrupole and so on. The dipole vanishes about the center of mass and thus the second term in the series, actually, corresponds to the quadrupole moment. In this series, each moment higher than the monopole, corrects the approximation for the cluster's asymmetry. Obviously, for practical computational purposes, the series evaluated to approximate the summation of interactions, must have a finite number of terms. The higher the number of terms considered, the smaller the error incurred.

An approximation using only the monopole, although convenient for its numerical simplicity, has the drawback of restricted accuracy. In Newtonian gravitation, it corresponds to assuming that the particles in a cluster are all symmetrically arranged around its center of mass. The less symmetry we observe in each cluster, the greater the error of the monopole approximation. When high accuracy is a goal and particle distributions are far from symmetrical, a more complex expression to approximate total interaction must be used.

Although the Barnes-Hut (BH) algorithm was originally developed for 3-D spaces, it is also lends itself to application in 2-D spaces like that of our simulation model. From now on, we discuss exclusively this monopole variant, which we used as proof of concept for this study.

The Barnes-Hut algorithm partitions the simulation space into square *cells*, which bear no relation to the hexagonal cells in our radio propagation model. These are, instead, an abstraction used to sub-divide the simulation space in a totally different way. Each cell can contain at most one particle: when a second particle is placed inside the cell, the space is recursively subdivided into smaller cells of equal dimension until each particle occupies a cell of its

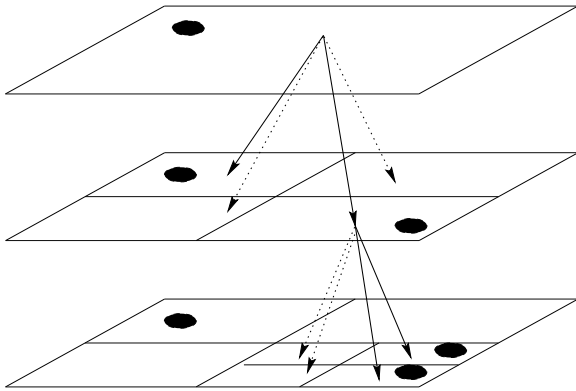


Figure 1. Evolution of a quad-tree as used in the Barnes-Hut algorithm

own. This defines a hierarchical organization of space; once a cell is subdivided it becomes the parent of the smaller cells it contains. Each cell is mapped to a node in a tree and each node contains a number of pointers, a mass value and a point coordinate.

The tree is constructed by a recursive procedure that starts with the definition of a cell big enough to contain the entire simulation space (initially empty). Particles are loaded onto the tree one at a time and at each addition a local subdivision of the space may occur. If any two particles are observed in the same cell, the space is further subdivided into four square sub-cells. The recursion ends when each particle finally occupies a different cell.

When we load the first particle onto the tree, it is added to the root cell. Then, as we add the second particle, we find that the root cell is not empty. Since a cell cannot contain more than one particle, we create four children for the root. If it so happens that, when we place the old and new particles in the newly created children, their positions allow them to fall inside different cells, this step is considered terminated. Proceeding similarly until all particles have been loaded, we obtain a quad-tree where all particles occupy leaf nodes. In the final structure no memory needs to be allocated for empty cells. Figure 1 shows the evolution of a tree as particles are added; each plane corresponds to a state after the addition of one more particle.

After the tree is constructed, each node must be tagged with two values: the multipole representing the “mass” of the cluster inside it and the spatial coordinate of its center of mass. Note that the same BH algorithm can be used inde-

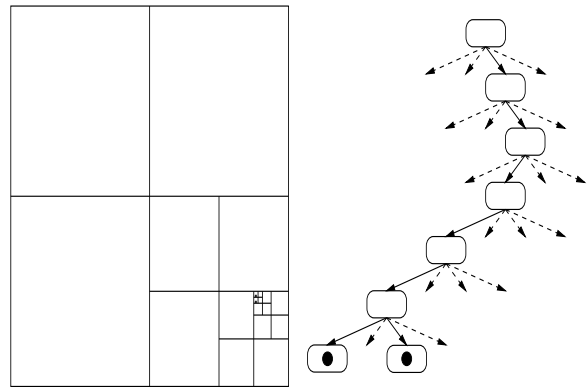


Figure 2. The Barnes-Hut tree can be sensitive to particle distribution

pendently of the quality of the approximation used to compute the multipole. This process starts down at the leaves: we make the node’s mass equal to the mass of the particle therein and the center of mass equal to the position of the particle. Then, we propagate these values one level up the tree: for each node we evaluate a truncated Taylor series considering the ensemble formed by its children and compute the new center of mass. The procedure is repeated for each level in the tree until the root node is reached.

Simulations with moving particles require that the tree be updated to reflect new arrangements of particles in space. Usually, at the beginning of each iteration, the tree is constructed from scratch and at the end it is destroyed once interactions have been computed. The costs of dynamic memory allocation are obviously non-trivial: depending on how fast these operations can be carried out at the programming language level, it may be beneficial to try out a few tricks, such as pre-allocating a large array of cells and using that pool instead of making repeated calls to the language’s operators. A more complex optimization would be to attempt to figure out how long the structure of the tree will be preserved: if the particles move slowly enough, the tree may not experience any change for a number of time-steps. This way, one would be able to keep the same tree for a while, updating only the particles’ coordinates.

An important fact to point out about this tree construction process is that the shape of the data structure is highly dependent on the particle distribution[1]. Consider the situation when we have two particles extremely close to each other. Applying the cell subdivision algorithm recursively, until each particle falls inside a different cell yields a very tall tree, as indicated in Figure 2. In many cases, including astrophysics simulations, this is the cause of serious per-

```

procedure BH_INTERACTION(p, ensemble)
begin
  if (singleton(ensemble))
    total = particle-particle interaction
  else
    if ((ensemble.diameter /
      distance(p,ensemble.centroid)) <  $\theta$ )
      total = particle-ensemble interaction
    else
      total =
        BH_Interaction(p, ensemble.child[0]) +
        BH_Interaction(p, ensemble.child[1]) +
        BH_Interaction(p, ensemble.child[2]) +
        BH_Interaction(p, ensemble.child[3])
    return total
end BH_INTERACTION

```

Figure 3. Barnes-Hut algorithm

formance degradation, since bodies can come indefinitely close to each other. The degradation comes not only from the tree construction process, but also from the added time taken traversing the data structure during the computation of interactions.

Once a tree has been constructed and tagged, the computation of total interaction over a given particle p follows the algorithm in Figure 3. The method determines whether it is necessary to compute individual interactions between a particle and an ensemble of particles using a *multipole acceptance criterion* (MAC) such as those defined in [17]. The original BH MAC is based on the ratio of the size l of the cell currently being inspected to the distance D between the cell's center of mass and p , what is called *opening angle*. The winning situation happens when, for p and some cell, $l/D < \theta$ (where θ is an accuracy parameter arbitrarily chosen). In this case the only interaction to compute is that of p and the cell, which is approximated by a single "virtual" particle of mass $ensemble.mass$ positioned at $ensemble.centroid$. If on the other hand $l/D > \theta$, then the algorithm is applied recursively to compute the interactions between p and the subdivisions of the ensemble.

We applied the BH algorithm with monopole approximations to the computation of interference noise in wireless simulations in a preliminary performance evaluation study. In our simulations, for each channel, we must compute noise in two different frequencies: the uplink and the downlink.

When we consider interference noise for uplinks, the particles in the algorithm are MSs. To accommodate multichannel systems, we must have one BH tree for each distinct channel: each tree collects all MSs transmitting on that

channel. Using the uplink trees, we compute the total interference incident on each BS receiver. Since we cannot have more than one user transmitting on the same channel inside the same cell (a wireless system cell, in this case), one could think users would always be separated by a minimum distance large enough to prevent the degeneration of the BH tree. However, there exists the possibility that users in the same channel, in neighboring cells, get close to the shared border and thus close to each other. This obviates the need for a tree construction method such as those proposed by Aluru [1].

For the downlink case, the situation is quite different. The BSs are the particles in the downlink BH trees and are well separated by definition: it would not make sense to put them close to each other because their coverage areas would experience too much overlap. In this case, the standard BH tree construction method can be used. Since bases are stationary objects, the trees can be constructed prior to the start of the simulation and their structure never changes (BSs are stationary). Moreover, there is no need for a distinct tree for each channel: the construction process takes into account only geographical positions to determine the tree topology and thus the shape of the data structure is the same for all channels. Instead of containing a scalar tag with multipole and center of mass, each node is associated with an array of tags, where each element corresponds to a different channel.

To follow our radio propagation model, both for the uplink and downlink, we must introduce a small difference from the conventional N -body problem. One of the factors in the numerator of equation 1 modifies path loss according to a random variable to represent transient propagation effects. When we evaluate interactions between a particle and an ensemble according to the BH algorithm, the random variable $L_{ij}(t)$ is sampled for a propagation path that is different than that of the brute-force algorithm. If we attempt to make a rigorous comparison between the interaction computed with the two methods, we may find out that the results are quite different; however, they are both statistically valid! In order to validate our models, in this paper, we have chosen to leave out the shadow fading factor. Once the real errors due to the approximations made in the BH algorithm are evaluated and the simulation results verified to within some arbitrary accuracy, this random variable can be reintroduced.

4 Experiments

We have experimented with three different methods to compute interference in a wireless cellular model: brute-force, Barnes-Hut and truncated (or limited) interference. In this paper, we limited our empirical analysis to the case of a single channel model. The networks we use are composed of $C \times C$ hexagonal cells with side lengths equal to 1000m

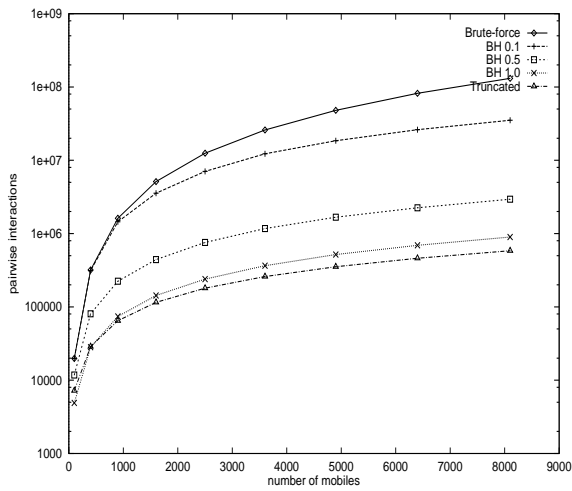


Figure 4. Number of pairwise interactions computed with Barnes-Hut, brute-force and limited interference

and the receivers are subject to internal noise $\nu = -120\text{dB}$. We suppose each cell has an active call, each mobile is placed randomly within its cell, uniformly at random on one of the three straight lines that bisect opposing sides of the hexagonal cell. This provides a “worst-case” scenario for BH, insofar as its computational efficiencies are achieved when the mobiles are sparsely separated. If significant performance gains are observed in this case, we will see better gains on networks with the same numbers of mobiles, distributed more sparsely over larger grids.

Varying the size of the network for $C \in \{10, 20, 30, 40, 50, 60, 70, 80, 90\}$ and, thus, also the number of mobiles in the system, we evaluated the total number of pairwise interactions (uplink and downlink) computed in each of the three methods. Figure 4 shows the results obtained (using a logarithmic scale on the y-axis) for three different values of opening angle θ in the BH algorithm. It is clear that the use of the BH algorithm gives the modeler the power to choose different levels of accuracy according to how much runtime can be afforded. Smaller values of θ bring the number of pairwise interactions computed up toward the same as in brute-force. In fact, for a sufficiently small θ , the BH algorithm will compute exactly the same values obtained with brute-force. It is for higher values of θ that BH becomes a really attractive option.

At $\theta = 1$, an entire cluster of particles is treated as a single one if the point where the interactions are measured is at least as far away from its center of mass as the BH cell containing it is large. The economy in the computation is so significant that interference can then be computed with

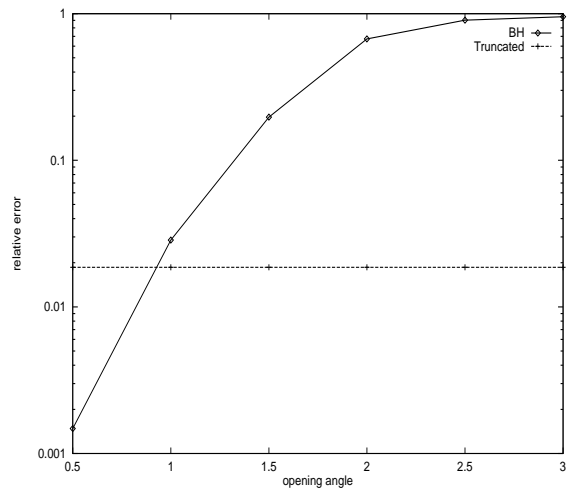


Figure 5. Mean relative error in downlink interference computation

a number of interactions in the same order of magnitude as with the limited interference method. The associated errors, however, are not negligible. Figure 5 shows that, for downlinks and $\theta = 1$, the mean relative error is about 50% higher than that of the truncated method (which does not consider the contribution of transmitters) within three times a cell-width. The situation is much worse for uplinks: as indicated in Figure 6 the mean relative error is about 300% higher. The reason for such disparity between these two cases perhaps lies in the diverse natures of spatial arrangement of transmitters and their interaction with the BH algorithm. In the computation of uplink interference, the particles in the BH tree are BS transmitters, which are arranged as a lattice. On the other hand, for downlinks, the arrangement of particles (MSs) on space is completely irregular.

For the same scenario, we have measured the execution time of a complete round of interference computations in the model on an SGI Origin 2000 with 180MHz clock, shown in Figure 7. This data helps quantify the cost of interference calculations. We see that for large mobile counts the execution improvement (about three orders of magnitude) corresponds with the reduction in numbers of interactions (about three orders of magnitude). This shows that indeed the added cost of building the quad-tree is amortized away on large models.

Nevertheless, there is no escaping the fact that in these experiments, simple truncation is faster, and more accurate than our implementation of BH. It is possible this is an artifact of the density of mobiles we employ in this experiment—BH will achieve higher levels of accuracy with less effort when the mobiles are farther separated. The error in the truncation method will diminish in this case

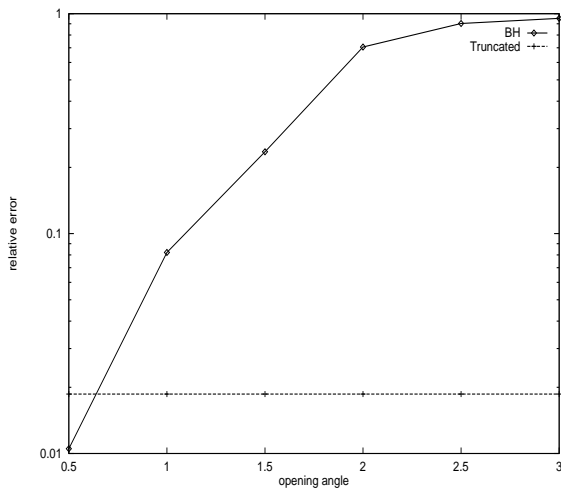


Figure 6. Mean relative error in uplink interference computation

also, but less so (having achieved most of its accuracy already). Clearly more experimental work is needed to evaluate this possibility.

5 Conclusions and Ongoing Work

Our preliminary investigation has shown that the application of N -body problem algorithms to the simulation of wireless cellular networks is promising. A substantial reduction in the size of interference computation was observed and the added scalability of these methods becomes crucial when we begin to consider bigger wireless models.

The errors produced with the BH algorithm, although controllable by an algorithm parameter, have proven to be larger than those of the limited interference (truncated) method. We have observed that to match the performance of the latter algorithm, BH has to be tuned to larger values of θ , which have the side-effect of aggravating the errors.

This fact is perhaps an indication that a more accurate variation of N -body algorithm may be more appropriate for wireless simulations. The use of higher order multipoles in the interference calculations could prove to be highly effective in decreasing these errors in spite of the extra floating point operations it would require. It may also be an artifact of the saturated system model we evaluated, and would not be seen in a model of a lightly loaded system. Our ongoing work is investigating such possibilities.

References

[1] S. Aluru, J. Gustafson, G. Prabhu, and F. E. Sevilgen. Distribution-independent hierarchical algorithms for the n -

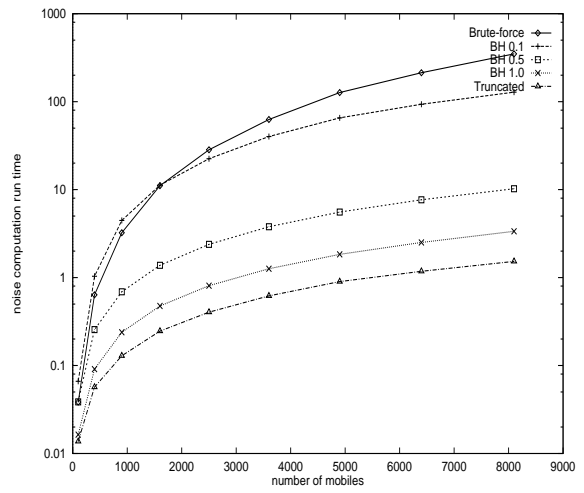


Figure 7. Performance comparison of Barnes-Hut, brute-force and limited interference

body problem. *Journal of Supercomputing*, 12(4):303–323, October 1998.

- [2] A. M. Appel. An efficient program for many-body simulation. *SIAM Journal of Scientific and Statistical Computing*, 6(1):85–103, January 1985.
- [3] Bajaj L., Takai M., Ahuja R., Tang K., Bagrodia R., and Gerla M. GloMoSim: A scalable network simulation environment. Technical Report 990027, UCLA, Computer Science Department, 1999.
- [4] J. Barnes and P. Hut. A hierarchical $O(N \log N)$ force-calculation algorithm. *Nature*, 324(4):446–449, December 1986.
- [5] J. E. Barnes. A modified tree code: Don't laugh; it runs. *Journal of Computational Physics*, 87:161–170, 1990.
- [6] S. Das, R. M. Fujimoto, K. Panesar, D. Allison, and M. Hybinette. GTW: A time warp system for shared memory multiprocessors. In *Proceedings of the 1994 Winter Simulation Conference*, pages 1332–1339, December 1994.
- [7] G. J. Foschini and Z. Miljanic. A simple distributed autonomous power control algorithms and its convergence. *IEEE Transactions on Vehicular Technology*, 40(4):641–646, November 1993.
- [8] G. J. Foschini and Z. Miljanic. Distributed autonomous wireless channel assignment algorithm with power control. *IEEE Transactions on Vehicular Technology*, 44(3):420–429, August 1995.
- [9] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73:325–348, 1987.
- [10] Kelly O., Lai J., Mandayam N., Ogielski A.T., Panchal J., and Yates R. Scalable parallel simulations of wireless networks with WiPPET: modeling of radio propagation, mobility and protocols. *IEEE International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'98)*, 1998.

- [11] Kelly O.E., Lai J., Mandayam N.B., Ogielski A.T., Panchal J., and Yates R.D. Scalable parallel simulations of wireless networks with WiPPET. *Wireless Networks*, 1999.
- [12] M. Liljenstam and R. Ayani. A model for parallel simulation of mobile telecommunication systems. In *4th International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS '96)*, pages 168–173, 1996.
- [13] M. Liljenstam and R. Ayani. Interference radius in pcs radio resource management simulations. In *Proceedings of the 1998 Winter Simulation Conference*, pages 1629–1637, 1998.
- [14] Liljenstam M. and Ayani R. Partitioning PCS for parallel simulation. *Proceedings Fifth International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 38–43, 1997.
- [15] J. Panchal, O. Kelly, J. Lai, N. Mandayam, A. T. Ogielski, and R. Yates. WiPPET: A virtual testbed for parallel simulations of wireless networks. In *Proceedings of the 12th Workshop on Parallel and Distributed Simulation (PADS '98)*, pages 162–169, 1998.
- [16] Panchal J., Kelly O., Lai J., Mandayam N., Ogielski A.T., and Yates R. Parallel simulations of wireless networks with ted: radio propagation, mobility and protocols. *Performance Evaluation Review*, 25(4):30–9, 1998.
- [17] M. S. Warren and J. K. Salmon. Astrophysical N-body simulations using hierarchical tree data structures. In *Supercomputing '92*, pages 570–576, Los Alamitos, 1992. IEEE Comp. Soc.
- [18] Zeng X., Bagrodia R., and Gerla M. GloMoSim: a library for parallel simulation of large-scale wireless networks. *Proceedings of PADS 98: Parallel and Distributed Simulation '98*, pages 154–61, 1998.