A Scalable Simulator for Forest Dynamics *

Sathish Govindarajan † Mike Dietze ‡ Pankaj K. Agarwal § James S. Clark ¶

Abstract

Models of forest ecosystems are needed to understand how climate and land-use change can impact biodiversity. In this paper we describe an individual-based, spatially-explicit forest simulator with full accounting of both landscape context and the fine-scale processes that influence forest dynamics. Unfortunately, performing realistic forest simulations of such models is computationally infeasible. We design efficient algorithms for computing seed dispersal and light, using a plethora of techniques. These include hierarchical spatial decomposition, monopole approximation and utilizing the graphics hardware for fast geometric computations. These algorithms allow us to simulate large landscapes for long periods of time.

Categories and Subject Descriptors: F.2.2 [Nonnumerical Algorithms and Problems]: Geometrical problems and computations

General Terms: Algorithms, Experimentation

Keywords: Forest models, Simulator, Ecological forecasting, Graphics hardware, Quad-tree, Monopole approximation.

1. Introduction

Forests cover approximately one-third of the Earth's land surface and account for 80% of terrestrial biomass [20]. Models of forest ecosystems are needed to understand how climate and land-use-change can impact biodiversity (i.e., the number and relative abundance of species), storage of carbon and forest resources at scales of decades to centuries. To be useful to scientists and managers, forest models must be sufficiently detailed to capture fine scale processes, such as establishment of seedlings in the forest understory, yet sufficiently broad to admit landscape and atmospheric processes.

Competition and *dispersal*, the two important spatio-temporal components of forest dynamics, have long frustrated efforts to simulate responses to global change. Competition for resources at fine spatial scales determines the species composition of forests. For example, light availability determines the growth rate of trees and their ability to survive. Species that can tolerate deep shade proliferate beneath dense forest canopies, whereas light demanding species exploit recently disturbed sites, which can be caused by fire, hurricanes, or land clearance. Intermediate light levels occur in "gaps", where death of one or a few large trees can create small openings in the forest canopy. Moreover, the light available to an individual depends on it's size – tall individuals experience a different light environment than do seedlings. Spatio-temporal variability, from small transient "sun flecks" to full canopy removal, makes computation expensive.

Seed production and dispersal determine local abundance and population spread across landscapes. The combination of large disturbances and changing climate means that species with abundant, well-dispersed seed can benefit when others may be threatened. Most seed falls close to the parent tree [5, 18], thus promoting aggregation. A fraction of seed that disperses long distance determines migration potential [6, 8]. Models cannot ignore tree size, as large individuals produce more seed than do small individuals. Moreover, species differences or "trade-offs" between long-distance dispersal capacity and competitive ability determine biodiversity [12, 14].

In this paper we describe an individual-based, spatially-explicit forest simulator with full accounting of both landscape context (1 sq. km.) and fine-scale processes that influence competition and dispersal. Individual-based models represent trees in terms of location, size, crown shape, species, age, growth rate, mortality risk, and reproductive ability. The landscape, in which the individuals are located, is heterogeneous with respect to the factors (e.g. light) that affect demographic rates. Our approach involves both a clarification of the computational issues and development of new algorithms that rely on approximation. The approximations make use of a comprehensive statistical treatment of variability and uncertainty that is parameterized from field data as basis for efficient simulation.

Related work. Models of forest dynamics, termed "stand simulators" or "gap models", have influenced ecological research for over 30 years [3, 16, 21, 22]. Early models focussed on light availability at intermediate spatial scales $(100m^2)$. These models were individual-based, but not spatially explicit [3, 21]. While allow-

*Work has been supported by NSF grants CCR-00-86013 EIA-98-70724, EIA-99-72879, EIA-01-31905, CCR-02-04118, BDEI-0131905 RR551-080-2401964, a DOE FACE grant and by a grant from the U.S.-Israeli Binational Science Foundation.

†Department of Computer Science, Box 90129, Duke University, Durham, NC 27708, USA. E-mail: gsat@cs.duke.edu

‡Department of Biology, Duke University, Durham, NC 27708, USA. E-mail: mcd7@duke.edu

§Department of Computer Science, Box 90129, Duke University, Durham, NC 27708, USA. E-mail: pankaj@cs.duke.edu

¶Department of Biology, Duke University, Durham, NC 27708, USA. E-mail: jimclark@duke.edu

SCG'04, June 8–11, 2004, Brooklyn, New York, USA.

ing analysis of relatively fine scale processes, such models miss the sub-grid variability that determines competitive interactions and the landscape 'connections' critical for understanding dispersal. More recent efforts treat the landscape as a uniform grid, modeling the dynamics within grid cells in a similar way as previous models, with the addition of neighbor-to-neighbor interactions [22]. SORTIE [16], the first spatially-explicit forest model, added spatially explicit light and dispersal. This approach allowed for spatial relationships at relevant scales, with limitations set by computational constraints to 9ha (300m x 300m). Due to the range of spatio-temporal scales involved, all effects have been limited by capacity to estimate parameters [16].

Our approach. Dispersal and light submodels form vital components of forest dynamics. They are spatial in nature, i.e, seed density and light availability at every point on the landscape depends on the location and geometric characteristics of all the trees in the forest. In fact, they are the primary limitation on simulating spatial models – the light calculation accounted for more than 90% of the runtime in the forest model of Pacala *et* al. [16]. Furthermore, this computational demand increases significantly with increasing landscape size.

Our forest model allows for many sources of variability and uncertainty that characterize processes and data. To compute all the processes exactly, we require $O(nA)$ calculations at each time step where n is the number of trees in the forest and A, the number of grid cells into which the landscape is partitioned. To perform efficient simulation, we balance accuracy against the stochasticity inherent in data and process. For example, uncertainty in light estimation is proportionately high at the lowest light levels. This means that precise computation at such light levels is unnecessary. In general, knowledge of uncertainty and error associated with parameterization guides the development of efficient algorithms.

Our contributions. We exploit spatial coherence to design efficient algorithms for dispersal calculations. We use quad tree [9, 19] to represent the forest at various spatial scales. Using the multi-resolution nature of the quad tree, we make spatial approximations, depending on the required accuracy. To compute dispersal, we use the monopole approximation [2] to aggregate seed dispersal from distant trees. This yields an efficiency-accuracy tradeoff scheme to compute dispersal. Experimental results show a speedup of about an order of magnitude for reasonable error. The graphics hardware is very efficient in performing certain basic primitives, simultaneously on all *pixels*. Recently, there has been some work on using the graphics hardware to solve geometric problems [13, 15, 17]. We exploit this parallelism of graphics hardware to obtain a hardware-based algorithm to calculate light. Our algorithm is at least two orders of magnitude faster than the naive algorithm.

All our experiments are performed on a 2.2 GHz Intel PC with 4 GB memory, nVidia Quadro4 XGL 900 graphic card running Linux OS.

Organization. We first describe our model and its components in Section 2. Section 3 describes briefly the various components of our model simulator. Then, in Section 4 and 5, we describe the algorithms that perform dispersal and light calculations and analyze their efficiency and accuracy. Finally, in Section 6, we present preliminary simulation results that indicate that the error in the computations have minimal impact on the dynamics of the model as a whole.

2. Forest Growth Model

The forest model consists of a landscape $\mathbb{L}$ and a population of trees. The landscape remains fixed, but the population changes with time. We first present an overview of the forest model. Then, we describe in detail the dispersal and light submodels.

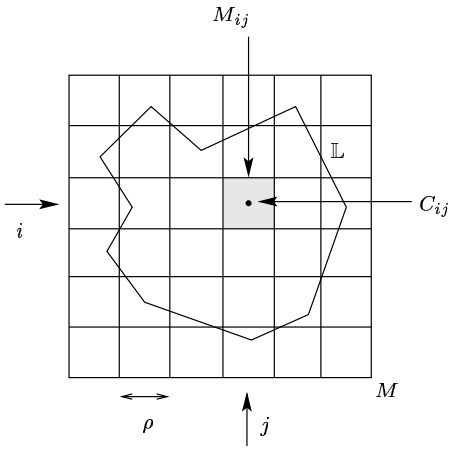

Figure 1: Landscape $\mathbb{L}$ and the underlying mesh M.

2.1 Overview of the Model

Landscape. Our model considers the landscape $\mathbb{L}$ of the forest as a planar polygonal region. The area of the landscape varies from a few hectares to few hundreds of hectares. We discretize $\mathbb{L}$ by enclosing it with a square and overlaying a uniform grid (mesh) M. Each grid cell M_{ij} of M is a square with side length ρ; we refer to ρ as the *resolution* of M. We use C_{ij} to denote the center of M_{ij}, and we associate an elevation (height) $z_{ij} \in \mathbb{R}$ with C_{ij}. By interpolating the heights at other points of $\mathbb{L}$, we can view $\mathbb{L}$ as a terrain. We can also associate various geological and urban features such as rivers, lakes, roads, etc. with $\mathbb{L}$. Figure 1 shows an example of a landscape alongwith the underlying mesh.

Population. Our model uses a hybrid representation of individual trees, depending on their size. The early stages of trees are modeled as densities, and after some growth, they are modeled as individuals with unique physical attributes. More precisely, we classify the population into five stages: *seed*, *yearling*, *seedling*, *sapling*, and *adult* (see Figure 2). We further refine the stage seed into *seed rain* and *seed bank* — the former representing the seeds that are dispersed by trees and the latter representing the ones that are on the ground. The seeds that have germinated are called yearlings. We model seed rain, seed bank, and yearling as densities, as they do not have any geometric attributes and all of them of the same species are identical. We assume that the density is uniform within each grid cell.

We model the next three stages — seedling, sapling, adult — as families of individuals. Each individual X has a physical location $\ell(X) \in \mathbb{R}^2$ and various physical attributes. Currently, we model each individual as a cylindrical trunk and a cylindrical canopy sitting on the trunk; see Figure 4. Let $D_t(X), H_t(X)$ denote the diameter and height of the trunk at time t. The diameter and the height of the canopy of X depend on $D_t(X)$ of X, and the relationship can be found in [7]. We define two "threshold" parameters: τ_D and τ_P. An individual X is a seedling if $H_t(X) \leq \tau_D$, a sapling if $\tau_D \leq H_t(X) \leq \tau_P$, and an adult if $H_t(X) \geq \tau_P$. Figure 3 shows the landscape with individuals at Duke Forest site.

Dynamics. The dynamics of our forest model consists of three parts — establishment of individuals, growth, and mortality. Indi-

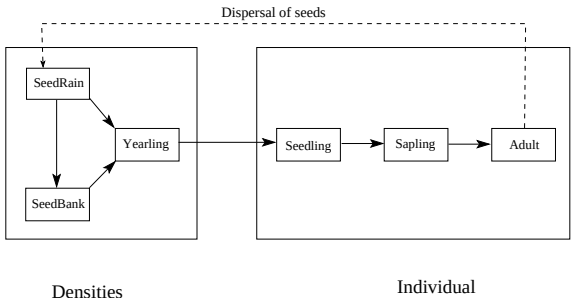

Figure 2: Evolution of densities of stages seed and yearling and growth of an individual from a seedling to an adult.

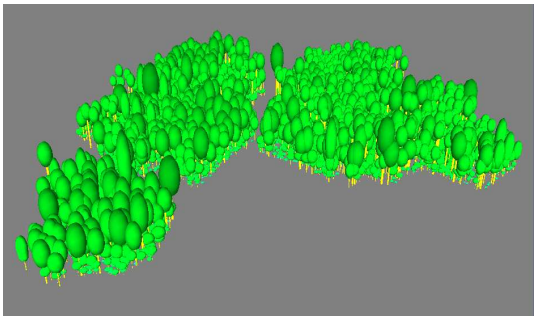

Figure 3: Landscape and individuals in Duke Forest site. Locations and trunk attributes (diameter and height) correspond to actual trees measured in meters.

viduals are established by dispersal of seeds. The adult trees produce seeds depending on $D_t(X)$ (also known as *diameter at breast height*) and these seeds are dispersed based on a dispersal kernel. The dispersal kernel accounts for both short and long distance dispersal. Growth of each of the stages is calculated based on resource availability and local density. Individuals are promoted from one stage to next based on the growth thresholds. An individual dies at the current time based on its *mortality probability*. The mortality probability is calculated based on the individual's growth suppression and natural disturbances.

Resources. The forest contains several resources like light, moisture, nitrogen, etc which are vital for the growth of an individual. We model each resource as a separate submodel. Light is considered as one of the main resources in our model. We have developed a sophisticated light model, based on Casetti's [4] light model, to calculate the availability of understory light at each grid cell.

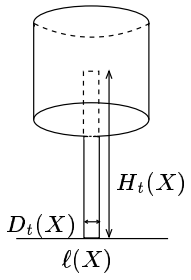

Figure 4: Geometric model of an individual.

2.2 Dispersal Submodel

The dispersal model determines the number of seeds that disperse into each grid cell of M. This quantity depends on:

- the number of seeds produced by each individual, denoted as *fecundity*.
- spatial distribution of seeds, which is defined by the *dispersal kernel*.

Fecundity. The reproductive output of an individual is nonzero only if it is a female and reproductively mature. The functional form of the fecundity, chosen from [7], is composed of factors that depend on the species to which that individual belongs and the size of the individual. The chosen form also includes a factor that captures the temporal variability. More precisely, $\beta_t(X)$, the fecundity of individual X at time t, has the following form:

$$\beta_t(X) = \chi(X) \cdot \Delta_t(X) \cdot 10^{a_0 + b(X) + \epsilon_t(X)} \cdot (D_t(X))^{a_1}, \quad (1)$$

where a_0 and a_1 are species-specific scaling parameters and $D_t(X)$ is the diameter of the trunk of individual X at time t. The functions $\chi(X)$ and $\Delta_t(X)$ are indicator functions, indicating the gender and reproductive maturity of X, respectively.

$$\begin{aligned} \chi(X) &= \begin{cases} 1 & \text{if } X \text{ is female,} \\ 0 & \text{if } X \text{ is male.} \end{cases} \\ \Delta_t(X) &= \begin{cases} 1 & \text{if } D_t(X) > \gamma(X), \\ 0 & \text{if } D_t(X) \le \gamma(X). \end{cases} \\ \gamma(X) &\sim \text{Gamma}(m_0, m_1). \end{aligned}$$

Here $\text{Gamma}(m_0, m_1)$ is the Gamma distribution with species-specific maturity parameters m_0, m_1. Next, $b(X)$ is an individual scaling parameter defined as:

$$b(X) \sim \text{Normal}(0, \tau^2).$$

$\text{Normal}(0, \tau^2)$ is the Normal distribution with species-specific parameter τ. Finally, $\epsilon_t(X)$ is a temporally autocorrelated Gaussian stochastic process, defined as:

$$\epsilon_t(X) \sim \text{Normal}(\nu \cdot \epsilon_{t-1}(X), \sigma^2),$$

where ν and σ are species-specific parameters.

Dispersal kernel. The dispersal kernel describes the spatial distribution of the scattering of seeds in the vicinity of the parent plant, as a function of distance r. We use a bivariate Student's t-distribution for the dispersal kernel, which has the following form:

$$f(r; u) = \frac{1}{\pi u \left[1 + r^2/u\right]^2}, \quad (2)$$

where u is a species specific parameter. Figure 5 shows the graph of the dispersal kernel for the parameters of two species: *Acer rubrum* and *Liriodendron tulipifera*.

Clark et al. [5] show that this kernel fits the dispersal data better than other kernels for most species. The kernel has a convex shape under the canopy of the individual which is responsible for short distance dispersal, and a fat tail which is responsible for long distance dispersal.

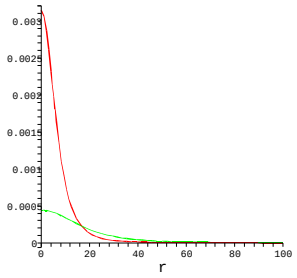

Figure 5: Dispersal kernel for parameters of species *Acer rubrum* u=101.3; *Liriodendron tulipifera* u=719.8. Parameter values are from Clark et al [7].

The actual number of seeds dispersed into the grid cell M_{ij}, denoted as s_{ijt}, is drawn from a Poisson distribution

$$s_{ijt} \sim \text{Poisson}(q_t(C_{ij}) \cdot \rho^2),$$

where ρ is the side length of grid cell M_{ij} and $q_t(y)$ is the expected seed density in location y at time t.

$$q_t(y) = \sum_X \beta_t(X) \cdot f(\|y - \ell(X)\|; u), \tag{3}$$

where the sum is taken over all the individuals in the forest.

2.3 Light Submodel

We now describe how we model the amount of light received by each individual, which in turn controls its growth. The amount of light received by a sapling or an adult is proportional to its exposed canopy area. A point p on the canopy of an individual is called *exposed* if the ray in $(+z)$-direction emanating from p does not intersect any other individual. The *exposed canopy area* (ECA), denoted by $A(X)$, is the area of the xy-projection of the exposed points on the canopy of X. The growth of a sapling or an adult X, denoted by $G(X)$, is measured by the change in its trunk diameter $D_t(X)$. It is calculated as follows:

$$G(X) = G_0 + G_1 \cdot \frac{A(X)}{K_A \cdot D_t(X)^{K_B} + A(X)},$$

where G_0, G_1 are the intercept and asymptote of growth rate and K_A, K_B are light saturation coefficients.

The amount of light received by a yearling or a seedling is proportional to the understory light at the grid cell that contains it. The *understory light* at a grid cell M_{ij}, denoted by L_{ij}, is the average annual intensity of sunlight that reaches M_{ij}, and is modeled as follows:

The spatial scale of our forest is sufficiently small, so we can assume that at any point of time, all points in the landscape receive the same intensity of light from any given direction. We model the sky as a positive hemisphere at infinity. We discretize this hemisphere into a uniform grid $\mathbb{H}$ by drawing latitude and longitude arcs. We refer to this grid as the *solar grid*. For a cell $H_{kl} \in \mathbb{H}$, let d_{kl} be the direction of the center of H_{kl} and I_{kl} be the annual sunlight intensity that emanates from H_{kl}. I_{kl} is computed using solar geometry calculations. Let $\mathbb{I} = [I_{kl}]_{k,l}$ be the intensity matrix, which will be computed in the preprocessing step. Figure 6 shows a photo taken at the Duke Forest site using wide angle lens. The photo was taken

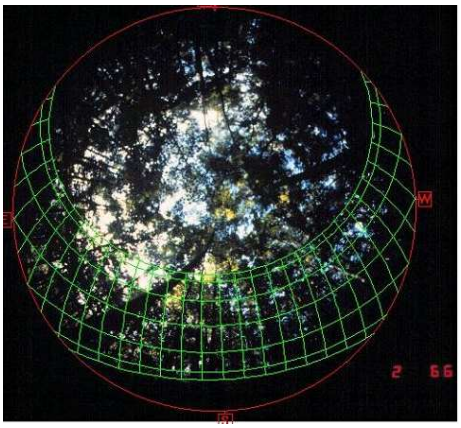

Figure 6: Canopy photo of adult individuals (as seen from the ground) at Duke forest site.

from the ground with the lens pointing in the $(+z)$-direction. The elevation and azimuth lines that are superimposed over the photo, correspond to regions of solar grid with high light intensity.

Let $r = r(i, j, k, l)$ be the ray emanating from C_{ij} in direction d_{kl}. Since, the solar grid is drawn on a hemisphere at infinity, r passes through the center of H_{kl}. Let $\mathbb{T}_r$ denote the set of saplings or adults individuals whose canopy intercepts r. Then the fraction of light that reaches grid cell M_{ij} is defined as

$$p_{ij}^{kl} = \begin{cases} 0 & \text{if } r \text{ intercepts trunk of any individual,} \\ \prod_{\tau \in \mathbb{T}_r} (1 - \lambda_\tau) & \text{otherwise.} \end{cases}$$

The amount of light in a tiny area dA in the neighborhood of C_{ij} and normal to d_{kl} is $I_{kl} \cdot p_{ij}^{kl} \cdot dA$. Therefore the light in a small area dA on the ground in the neighborhood of C_{ij} is $I_{kl} \cdot p_{ij}^{kl} \cdot \cos(\theta_k) \cdot dA$, where θ_k is the elevation angle of d_{kl}. The understory light L_{ij} is now defined as

$$L_{ij} = \sum_{k,l} I_{kl} \cdot p_{ij}^{kl} \cdot \cos(\theta_k). \tag{4}$$

The growth of a yearling or seedling X, denoted by $G(X)$, is measured by the change in its height $H_t(X)$. It is calculated as follows:

$$G_X = G_0 + G_1 \frac{L_{ij}}{K + L_{ij}} \cdot e^{-(\zeta D_{con} + \zeta_h D_{hetero})}.$$

Here, L_{ij} is the understory light in the grid cell containing the individual X. G_0, G_1 are the intercept and asymptotes of growth rate, K is the light saturation coefficient, D_{con} (resp. D_{hetero}) is the number of individuals of same (resp. different) species present in the grid cell containing X, and ζ, ζ_h are density dependent coefficients.

3. Our Model Simulator

We have developed a forest simulator based on the model outlined in Section 2. Submodels include dispersal, light, germination and mortality. The simulator takes as input an initial configuration of the forest and landscape. It simulates dynamics at annual time steps. Figure 7 shows the flowchart of operations.

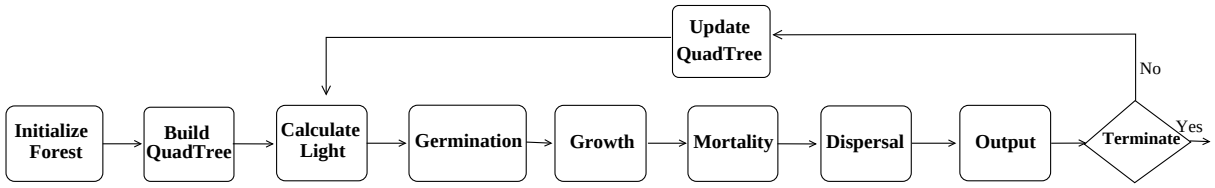

Figure 7: Flow chart of the sequence of operations performed by the simulator.

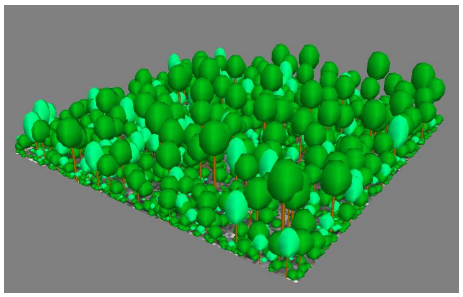

Figure 8: Snapshot image of the forest simulated by our model.

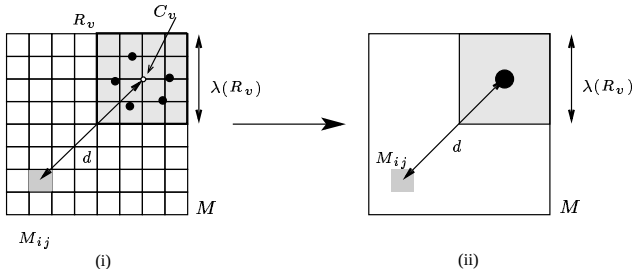

Figure 9: Monopole approximation for dispersal. (i) A 8×8 mesh M, grid cell M_{ij} (shaded) and a 4×4 region containing 5 individuals (shown as small circles). (ii) The mesh after monopole approximation. The "super individual" (shown as a large circle) is located at the center of the 4×4 region.

Since dispersal and light calculations are computationally intensive, and ecological experiments need to be performed on large landscapes (at least 1 sq. km.) and for long durations (up to several thousand years), performing exact calculations to simulate dynamics would take months (e.g. a time step on 512×512 landscape took 5 hours). We therefore expedite the simulation by performing calculations approximately — the approximation error can be controlled by the user (the approximation error is fixed such that it is within the inherent stochasticity of the model). We maintain a hierarchical (multi-resolution) representation of the forest using a quad-tree data structure and calculate dispersal approximating at spatial resolutions depending on the required accuracy. A hardware-based algorithm is used to calculate light. The dispersal and light algorithms are explained in Section 4 and 5, respectively.

The simulator outputs the densities of seed banks and yearlings for each grid cell and the locations and characteristics of all the individuals. We have developed a visualization package using OpenGL/GLUT, which allows the user to view forest dynamics and to navigate through the forest interactively. The user can zoom-in/zoom-out, controlling both view point and viewing direction. Figure 8 shows a snapshot image of the visualization of the forest simulated by our model. We also implemented another visualization package to view density data (seed bank and yearlings) together with summary statistics, using Amira [1], a volume-rendering package.

4. Computing Dispersal

In this section, we describe our dispersal algorithm. We first describe the quad-tree based data structure and then we describe the approximation algorithm to calculate dispersal. In our descriptions, we assume, for sake of simplicity, that all individuals in the forest belong to the same species. The data structure and algorithm can be extended to the case of multiple species in an obvious manner. Finally, we present experimental results that show the performance of our algorithm.

Quad-tree data structure. For simplicity, we assume that $\mathbb{L}$ is enclosed in a square of side-length 2^l and discretized into a $2^l \times 2^l$ mesh M, where $l \geq 0$ is an integer; we assume that $\rho = 1$. Let A denote the area of $\mathbb{L}$, which is also the number of grid cells of M. We denote by I, the set of all the individuals (saplings, adults) in the forest and by P, the set of their locations.

$$P = \{\ell(X) \mid X \in I\}.$$

A quad-tree T on $\mathbb{L}$ is a 4-way tree that represents a hierarchical subdivision of $\mathbb{L}$. Each node v of T is associated with a square $R_v \subseteq \mathbb{L}$, a subset $P_v = R_v \cap P$ of points, and a set $I_v = \{X \mid \ell(X) \in R_v\}$ of individuals. For the root u of T, $R_u = \mathbb{L}$, $P_u = P$ and $I_u = I$. If R_v is a grid cell of M or $|P_v| = 1$, v is a leaf. Otherwise, we partition R_v into four congruent squares by bisecting its two sides, and assigning the four squares to the four children of v. If the depth of a node v is d, then the side-length of R_v is 2^{l-d}. The maximum depth of T is $\log_4 A$.[1]

In each node v, we store $|P_v|$, the total number of individuals and $b_t(v) = \sum_{X \in I_v} \beta_t(X)$, the total fecundity of all the individuals in I_v. (If we have multiple species, we store this information for each species separately). We use this information to develop an approximation scheme to compute dispersal. At each leaf v of T, we store the sets I_v and P_v in a list.

Approximation scheme. The dispersal algorithm computes the expected number of seeds $q_t(C_{ij})$, from all the individuals of the forest, that fall into each grid cell M_{ij} at time t using (3). The exact computation takes $O(nA)$ time since we iterate over all grid cells and individuals. This is too expensive even for a moderately sized forest. For example, it takes about four hours to compute dispersal exactly on a 1024×1024 landscape with 500,000 individuals.

We rely on an approximation scheme to expedite the computation at a slight loss in accuracy. We first describe the intuition and

[1] In general, a node v of a quad-tree is a leaf only if $|P_v| = 1$, but in our application it suffices to stop the subdivision as soon as we reach a grid cell. This ensures that the depth of T is $\log_4 A$.

Algorithm 1 `Monopole Approximation`(v, M_{ij})

C_v: center of square R_v
if $(\lambda(R_v)/\|C_v - C_{ij}\| \leq \mu)$ **then**
  return $\beta_t(X_v) \cdot f(\|C_{ij} - C_v\|; u)$
else
  if v is a leaf **then**
    return $\sum_{X \in I_v} \beta_t(X) \cdot f(\|C_{ij} - \ell(X)\|; u)$
  else
    for each child w of v **do**
      return `Monopole Approximation`(w, M_{ij})

then give a formal description. It is clear from (2) and (3) that if an individual is far away from a grid cell M_{ij}, then the expected number of seeds falling into M_{ij} is almost the same if we vary the location of X a little; see Figure 9. We therefore cluster the grid cells that are far away from M_{ij} and move all individuals in a single cluster to a canonical location. We regard all these individuals as a single "super individual" X_v, whose fecundity $\beta_t(X_v)$ is the sum of the fecundity of all the individuals in the cluster. The quad tree provides a natural way of computing this clustering. We refer to this approximation as *monopole approximation*.

For a node v of T, let C_v (resp. $\lambda(R_v)$) denote the center (resp. side-length) of R_v. See Figure 9. We set a threshold parameter μ called *monopole coefficient*. If $\lambda(R_v)/\|C_v - C_{ij}\| \leq \mu$ (grid cell M_{ij} is far away from R_v as compared to side length of R_v), we perform the monopole approximation, i.e. replace all the individuals I_v with the "super individual" X_v located at C_v. The seeds falling into M_{ij} due to individuals I_v is approximated by the seeds falling into M_{ij} due to X_v.

We now describe the algorithm formally. It is a recursive procedure, starting at the root of the quad tree. We perform the following at each node v of T: we check whether the monopole approximation can be performed at v i.e. $\lambda(R_v)/\|C_v - C_{ij}\| \leq \mu$. If so, we approximate the expected number of seeds falling into M_{ij} due to individuals I_v by calculating the expected number of seeds falling into M_{ij} due to the "super individual" X_v. If $\lambda(R_v)/\|C_v - C_{ij}\| > \mu$, and v is a leaf, we calculate the expected number of seeds falling into M_{ij} due to individuals in I_v by summing the contribution from each individual in I_v. Else, we recurse on the children w of v. Algorithm 1 gives the pseudo code for the algorithm.

We perform the Monopole procedure at each grid cell M_{ij}. It can be shown that the time complexity of the above algorithm is $O(A \log A + n)$.

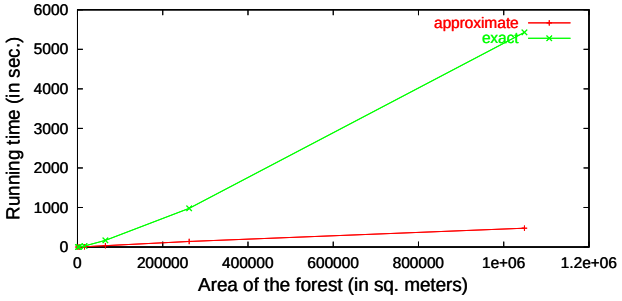

Figure 10: Running time of dispersal algorithm with monopole 0.1 for varying forest area sizes.

Experimental results. We report the results of a set of experiments performed on our forest simulator. Here we emphasize the performance of the approximation algorithm for forests of different areas A and different monopole coefficients μ. We measure the performance in terms of running time and RMS value of relative error in dispersed seeds. For a more detailed study on the performance of our dispersal algorithm, see [10]. Let s_{ij} denote the actual seed rain that falls in grid cell M_{ij}, as computed by the exact algorithm and $\tilde{s}_{ij}$ denote the seed rain in M_{ij} as computed by the approximation algorithm. We define the relative error ε_{ij} to be

$$\varepsilon_{ij} = \left| 1 - \frac{\tilde{s}_{ij}}{s_{ij}} \right|.$$

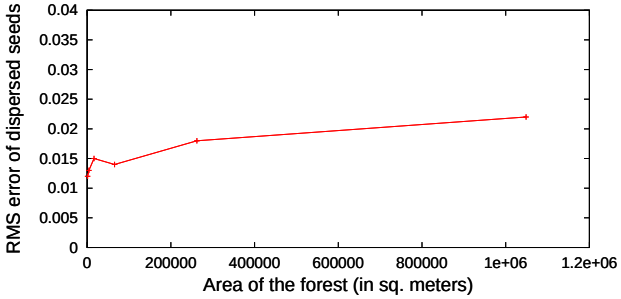

Figure 11: RMS relative error of approximation algorithm with monopole 0.1 for varying forest area sizes.

The RMS value of ε_{ij} is defined as:

$$E = \left(\frac{1}{A} \sum_{i,j} \left(1 - \frac{\tilde{s}_{ij}}{s_{ij}} \right)^2 \right)^{1/2}.$$

We performed experiments on a forest initialized with the output of a 100 year simulation involving a single species. In our experiment, we varied the side length of the forest from 32 meters to 1024 meters. Figure 10 compares the running time of the exact algorithm with the approximation algorithm (for monopole coefficient 0.1). The exact algorithm is the monopole algorithm with monopole coefficient set to 0. For a 512×512 sq. m forest, the monopole achieves speedup of two orders of magnitude. Figure 11 plots the value of E, for monopole coefficient 0.1. Note that E, the RMS error in seeds dispersed, is less than 2% for the landscapes simulated.

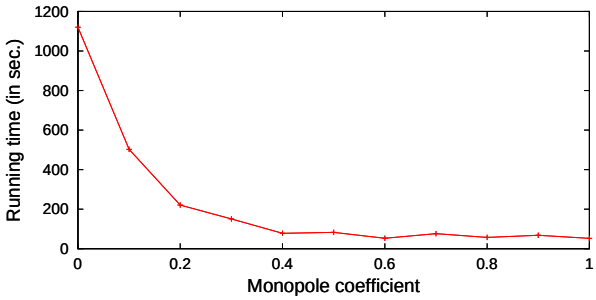

Figure 12: Plot of running time of monopole algorithm on a 512×512 landscape with varying monopole coeffi cient.

Next we performed experiments on a forest with side length 512 meters, with monopole coefficient ranging from 0.1 to 1.0. Figure 12 and 13 shows the running time and the value of E for different monopole coefficient. As anticipated, the run time decrease

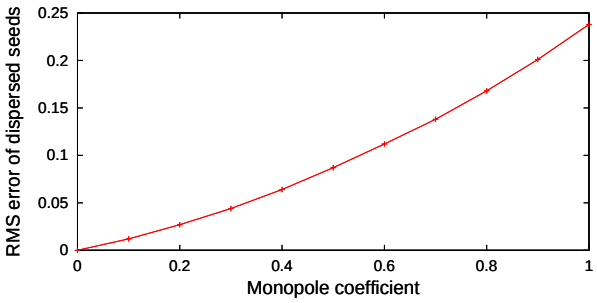

Figure 13: Plot of RMS error of monopole algorithm on a 512×512 landscape with varying monopole coeffi cient.

with the increase in the monopole coefficient. Similarly, the RMS error increases with increase in the monopole coefficient.

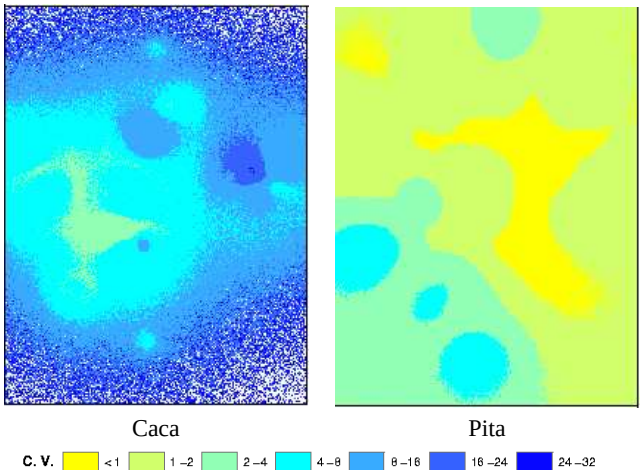

Figure 14: Spatial map of coeffi cient of variation in the number of seeds dispersed for species *Caca* and *Pita*.

The next experiment compares the inherent variability of the dispersal process with the relative error incurred by the approximation algorithm. First, we quantify the inherent variability (stochasticity) of the dispersal model using statistical methods. We perform $N = 1000$ iterations of dispersal computation on the Duke Forest stand, located in the Blackwood Division of the Duke Forest in Chapel Hill, NC. In the Duke Forest stand, every individual over 2m tall was identified to species, mapped, and its diameter was measured at 1.45m high, a common metric in forestry and ecology referred to as Diameter Breast Height (DBH). In total there were 52 species observed in the stand, but in this paper we will focus on two species: *Carpinus caroliniana* (CAca) and *Pinus taeda* (PIta). The Duke forest stand was approximately centered in a 512×512 landscape at $1m \times 1m$ resolution. We set the monopole coefficient to 0.125 so that the error in computation is negligible ($\leq 2\%$).

For each iteration, the simulator outputs the spatial map of seed rain, number of seeds dispersed, in the forest. Let s_{ijt} denote the seed rain at grid cell M_{ij} in iteration $t, 1 \leq i, j \leq 512, 1 \leq t \leq N$. From the N replicate dispersal maps, we calculate the mean seed rain $\hat{s}_{ij}$ at grid cell M_{ij} using the formula:

$$\hat{s}_{ij} = \frac{1}{N} \sum_{t=1}^{N} s_{ijt}.$$

We also calculate the *coefficient of variation* of seed rain CV_{ij}, using the following formula:

$$CV_{ij} = \left(\frac{1}{N} \sum_{t=1}^{N} \left(\frac{s_{ijt}}{\hat{s}_{ij}} - 1 \right)^2 \right)^{1/2}.$$

Figure 14 shows the spatial pattern of the temporal variability (CV) in seed rain for species *Caca* and *Pita*. For each species, the RMS value of CV_{ij}, denoted by CV, captures the inherent variability of the dispersal process for that species. The following formula expresses CV in terms of seed rain s_{ijt} and mean seed rain $\hat{s}_{ij}$:

$$CV = \left(\frac{1}{N \cdot A} \sum_{i,j,t} \left(\frac{s_{ijt}}{\hat{s}_{ij}} - 1 \right)^2 \right)^{1/2}.$$

The CV value for species *Caca* and *Pita* are 65.12 and 1.45 respectively. This indicates that species Caca has high inherent variability as compared to species Pita.

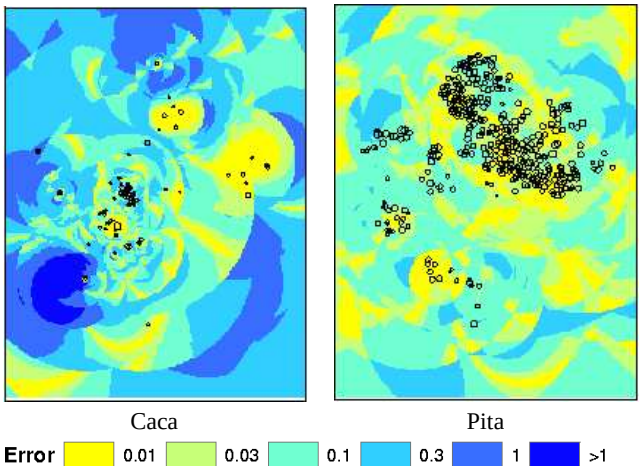

Figure 15: Spatial map of relative error for species *Caca* and *Pita*. Monopole coeffi cient is set to 0.5.

Next, we study how the relative error ϵ_{ij} varies with the monopole coefficient. Figure 15 shows the spatial distribution of ε_{ij} with $\mu = 0.5$ for species *Caca* and *Pita*. The circular arc patterns in the figures is an artifact of our algorithm — the portion of the forest in which we perform the monopole approximation at the same level of the quad-tree is bounded by a circle.

The RMS relative error for species *Caca* and *Pita* for $\mu = 0.5$ are 0.25 and 0.05 respectively. The RMS relative error is at least an order of magnitude smaller than the inherent variability, implying that the error incurred by the approximation is quite tolerable.

5. Computing Light

In this section we describe our algorithm to calculate light. First, we briefly describe our algorithm to compute exposed canopy area (ECA). Then, we describe the graphics hardware-based algorithm to compute understory light. Finally, we provide experimental results to demonstrate the efficiency of our algorithm.

5.1 Computing exposed canopy area

The algorithm computes for each sapling or adult X, the area of the xy-projection of the exposed points of the canopy of X. Let $\mathbb{T}$

denote the set of all the adults and saplings in the forest. Since, we model the canopy of X as a cylinder, its xy-projection is a circle $C(X)$. Let $h(X)$ be the height of the top of the canopy of X from the ground. The main idea of the algorithm is to assign each grid cell M_{ij} to the tallest individual $X \in \mathbb{T}$, such that $C(X)$ contains M_{ij} completely. If there exists no individuals $X \in \mathbb{T}$ such that its canopy projection $C(X)$ completely contains M_{ij}, we leave grid cell M_{ij} unassigned. Our algorithm performs this assignment by considering individuals $X \in \mathbb{T}$ in decreasing order of $h(X)$, and assigning to X, all the grid cells that are completely contained in $C(X)$ and have not been assigned already. It is easy to see that the exposed canopy area $A(X)$ of X can be approximated by the sum of the areas of all the grid cells assigned to X. We can increase the accuracy of the ECA calculation by using a mesh with finer resolution.

5.2 Computing understory light

First we describe the exact algorithm to compute understory light at each grid cell M_{ij} of M. Then, we describe an efficient graphics hardware-based algorithm to compute understory light.

Exact algorithm. We calculate the understory light at grid cell M_{ij} as follows: Let τ be a sapling or adult in the forest. For each direction d_{kl}, we determine the fraction of light, f_{ij}^{kl}, that reaches M_{ij} (after attenuation by τ). Let $r = r(i, j, k, l)$ be the ray emanating from C_{ij} in direction d_{kl}.

$$f_{ij}^{kl} = \begin{cases} 0 & \text{if } r \text{ intercepts trunk of } \tau, \\ (1 - \lambda_\tau) & \text{if } r \text{ intercepts canopy of } \tau, \\ 1 & \text{otherwise.} \end{cases}$$

The calculation of this fraction f involves deciding whether the ray r intersects τ. We repeat this calculation for all the saplings and adults in the forest. Now, $p_{ij}^{kl} = \Pi_\tau f_{ij}^{kl}$. We then calculate the understory light L_{ij} using (4).

The above algorithm took 4 hours to compute understory light on a 512×512 landscape having 50,000 individuals.

A graphics hardware-based algorithm. The light model calculation, as described in Section 2.3, is similar to the visibility computation in graphics hardware. We are given a view point (grid cell M_{ij}) and a collection of geometric objects (individuals). The light model requires a set of directions d_{kl} that are visible from the view point. We perform this calculation for all the grid cells. One of the primary capabilities of the *graphics hardware* is to perform such visibility computations simultaneously for all grid cells in M. Since the number of grid cells in our forest is large (about 10^6 grid cells), the graphics hardware provides an efficient method for light computation. The main challenge in using the graphics hardware for calculating light is the representation of the translucence of individuals. Light intercepting the canopy of an individual τ is attenuated by a factor of λ_τ. To perform light-object interactions using hardware requires a simple form of volume rendering.

The algorithm has the following structure:

1. For a given direction, calculate attenuation fraction p_{ij}^{kl} for all grid cells M_{ij} using the graphics hardware.
2. Repeat above step for all directions d_{kl}.
3. Calculate I_{ij} for all grid cells M_{ij} using (4).

We observe the forest from infinite distance along angle d_{kl} and use the hardware color accumulation operation of color blending to compute the product p_{ij}^{kl} for all grid cells M_{ij} of M. Because the only operations allowed in color blending are $+$, $-$, min, and max, we use a simple trick, namely, compute $\log p_{ij}^{kl} = \sum_\tau \log(1 - \lambda_\tau)$. The graphics hardware calculates $\log p_{ij}^{kl}$ using color blending, from which we can compute p_{ij}^{kl}.

The steps needed to compute p_{ij}^{kl} are:

1. **Setting light source and image plane**: Fix the light source at infinity along direction d_{kl}. Set the image plane as a plane orthogonal to d_{kl} such that all the individuals are in between the source and image plane.
2. **Render individuals**: Draw each individual as geometric objects (canopy and trunk are drawn as cylinders of appropriate diameter and height). The color of the canopy cylinder is set to $\log(1 - \lambda_\tau)$ (for blending) and the color of trunk is set to $\log \epsilon$, where ϵ is an arbitrarily small constant. Since ϵ is small, almost all the light is blocked by the trunk.
3. **Read color buffer**: The color buffer contains the result of the color blending operation for all pixels in the image plane.
4. **Projection**: Our objective is to calculate light at the grid cells, which lie on a horizontal plane. The color buffer contains light on pixels of the image plane that is oblique to the horizontal plane. We perform a projection operation to project each pixel on image plane to the appropriate pixel on horizontal plane.

From experimental results, it is found that the projection operation is costly, taking about 25% of the total time.

We optimize the algorithm by eliminating the projection. The need for projection arises because the image plane is oblique to the source direction. We can fix the source along the vertical direction and transform the individuals such that one obtains the same result. The transformation depends on the direction of light source. For a given direction d_{kl} and an individual τ, the grid cells that receive attenuated light due to τ are those that form the shadow of τ when light comes from direction d_{kl}. Thus, for a given direction, we compute the shadow of individual τ and render the shape of the shadow. For vertical cylinders, the shadow is a rectangle with a semi-disk at each end and can be easily computed. Figure 16 (i) shows an individual τ rendered as a cylinder by the original algorithm. Light rays along direction d_{kl}, intercept τ and form the shadow $\phi_{kl}(\tau)$. Figure 16 (ii) shows the shadow $\phi_{kl}(\tau)$ rendered directly by the optimized algorithm.

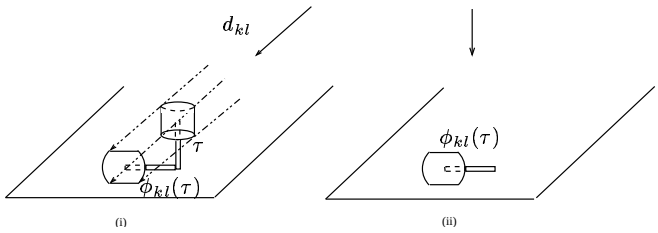

Figure 16: (i) Algorithm renders individual τ as a cylinder. Light rays that come along direction d_{kl} intercepts τ and forms a shadow $\phi_{kl}(\tau)$. (ii) Optimized algorithm renders the transformed individual (shadow) $\phi_{kl}(\tau)$. Light rays come from the vertical direction.

Experimental results. We evaluated the performance of the hardware algorithm with a set of experiments where we varied the side

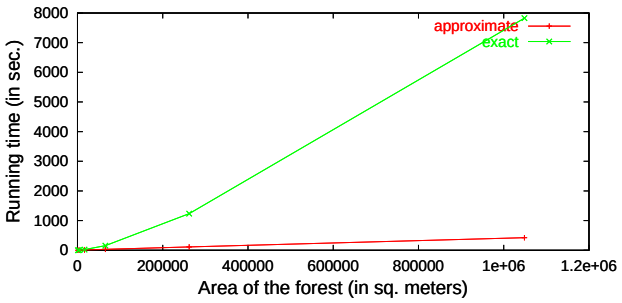

Figure 17: Comparison of running time of exact algorithm and hardware algorithm for varying forest areas.

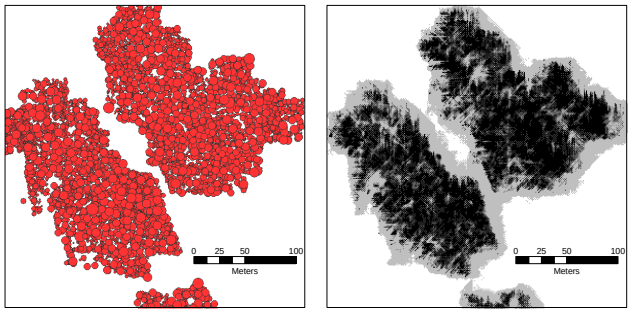

Figure 18: (a) The canopy of the adults and saplings (represented by circles) in Duke forest site. (b) The light intensity map calculated by our model for these adults and saplings.

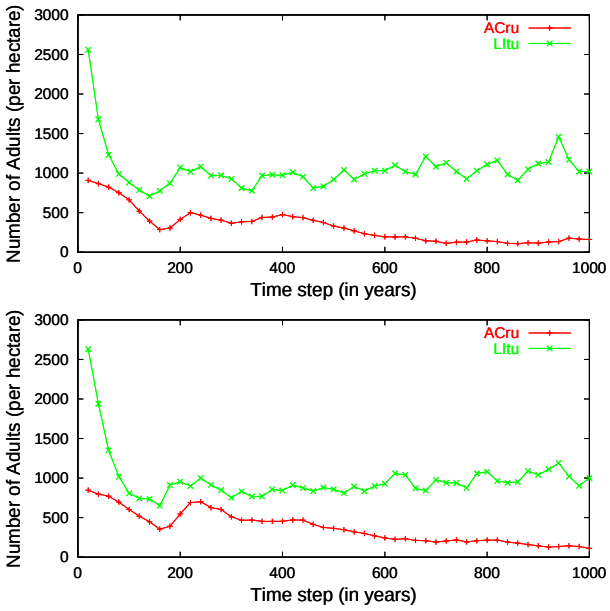

Figure 19: Comparison of coexistence experiment with 2 species for (a) exact dispersal and (b) approximate dispersal (μ = 0.125)

length of the forest from 32 meters to 1024 meters. Figure 17 compares the running time of the hardware algorithm and the exact algorithm to calculate light. For a 1024×1024 landscape, the hardware algorithm is at least two orders of magnitude faster than the exact algorithm.

Figure 18 (a) shows the top view of the adults and sapling canopy in Duke Forest site. The canopies are depicted as circles with the actual canopy radius of the individual. Figure 18 (b) shows the light intensity map calculated by our model for these adults and saplings. The light calculation captures quite well, the spatial variability in understory light that results from variability in individual size and location.

6. Ecological Experiments

To determine if the approximation algorithms have an impact on modeled forest dynamics, we present two 1000 runs of the forest model on a 64×64 landscape. These runs illustrate the competitive dynamics of two species, *Acer rubrum* and *Liriodendron tulipifera*. In both the runs, the landscape was initialized to identical conditions. One of the runs used the exact dispersal algorithm, and the other, the approximate algorithm with $\mu = 0.125$. Both the runs used graphics hardware to calculate light. As shown in Figure 19, when operating within a reasonable error bound, the impact of approximation of model dynamics is negligible when compounded over the long term. Furthermore, the runs illustrated in Figure 19 include numerous sources of uncertainty and stochasticity so any differences between runs is hard to attribute to computational error alone.

In addition to accuracy, it's essential that the forest simulator provide ecologists with a model that is sufficiently fast to use for experiments. We performed two runs for the same experiment as

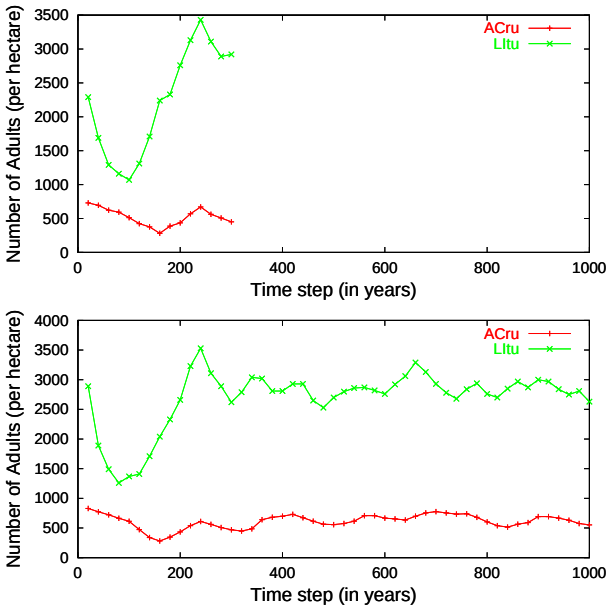

Figure 20: Comparison of coexistence experiment on 512×512 landscape. (a) exact dispersal calculation for 300 years and (b) approximate dispersal calculation (μ = 0.125) for 1000 years.

above on a 512×512m landscape. In both the runs, the landscape was initialized to identical conditions. One of the runs performed exact dispersal calculation, while the other performed approximate dispersal calculation with $\mu = 0.125$. Both the runs used graphics hardware to calculate light. The exact algorithm took about one and a half hours to calculate one time step of exact dispersal calculation. We ran the exact algorithm for two weeks and it could only complete 300 years of simulation. On the other hand, the approximation algorithm completed 1000 years of simulation in about two days.

7. Discussion

We have developed efficient algorithms to simulate an individual based, spatially explicit forest model. The speed-up of our simulator over a naive calculation was made possible by the following algorithms:

- Monopole based approximation algorithm to calculate dispersal. Experiments shows a speedup of an order of magnitude for reasonable error.
- A novel graphics hardware-based algorithm for calculating understory light. Our algorithm is two orders of magnitude faster than the naive method.

Work presented in this paper is part of an ongoing inter-disciplinary project to study forest ecosystems using simulation. We have began using our simulator to perform experiments that address ecological issues like coexistence and migration.

Some of the interesting algorithmic issues and future direction include:

- Our current dispersal algorithm has complexity O($n \log n$). We plan to adapt the linear time *Multipole algorithm* of Greengard [11] to calculate dispersal.
- We plan to develop a better light model that captures the temporal variability of light intensity. Based on this model, we plan to design efficient hardware-based algorithms to compute light.
- Extend our model to include terrain features like roads, lakes, buildings, etc.
- Extend our model to include additional environmental factors such as soil moisture, nitrogen and disturbance agents such as fire and wind.

References

[1] Amira - an advanced 3d visualization and volume modeling system, 2001.

[2] J. Barnes and P. Hut, A hierarchical $O(n \log n)$ force-calculation algorithm, *Nature*, 324 (1986), 446–449.

[3] Botkin, Janak, and Wallis, Some ecological consequences of computer model of forest growth, *J. Ecology*, 60 (1972), 101–116.

[4] A. Cescatti, Modelling the radiative transfer in discontinuous canopies of asymetric crowns. i. model structure and algorithms, *Ecol. Modeling*, 101 (1997), 263–274.

[5] J. Clark, M. Silman, R. Kern, E. Maclin, and J. HilleRisLambers, Seed dispersal near and far: patterns across temparate and tropical forests, *Ecology*, 80 (1999), 1475–1494.

[6] J. S. Clark, C. Fastie, G. Hurtt, S. T. Jackson, C. Johnson, G. King, M. Lewis, J. L. J, S. Pacala, I. C. Prentice, W. Schupp, T. Webb, and P. Wyckoff, Reid's paradox of rapid plant migration, *BioScience*, 48 (1998), 13–24.

[7] J. S. Clark, S. LaDeau, and I. Ibanez, Fecundity of trees and the colonization-competition hypothesis, in press (2004).

[8] J. S. Clark, M. Lewis, and L. Horvath, Invasion by extremes: variation in dispersal and reproduction retards population spread, *American Naturalist*, 157 (2001), 537–554.

[9] R. A. Finkel and J. L. Bentley, Quad trees: a data structure for retrieval on composite keys, *Acta Inform.*, 4 (1974), 1–9.

[10] S. Govindarajan, M. Dietze, P. K. Agarwal, and J. Clark, A scalable algorithm for dispersing population, *Accepted. In Special Issue of Jounal of Intelligent Information Systems*.

[11] L. Greengard, *The rapid evaluation of potential fields in particle systems*, MIT Press, Cambridge, 1988.

[12] A. Hastings, Disturbance, coexistance, history, and competition for space., *Theoretical Population Biology*, 18 (1980), 363–373.

[13] K. E. Hoff, III, T. Culver, J. Keyser, M. Lin, and D. Manocha, Fast computation of generalized Voronoi diagrams using graphics hardware, *Proc. ACM SIGGRAPH*, 1999, pp. 277–286.

[14] C. G. Hurtt and S. W. Pacala, The consequences of recruitment limitation: reconciling chance, history, and competative differences between plants., *J. theor. Biol*, (1995), 1–12.

[15] N. Mustafa, E. Koutsofios, S. Krishnan, and S. Venkatasubramanian, Hardware-assisted view-dependent map simplification, *Proc. 17th ACM Symposium on Computational Geometry*, 2001, pp. 50–59.

[16] S. W. Pacala, C. D. Canham, and J. A. S. Jr., Forest models defined by field measurements: 1 the design of a northeastern forest simulator, *Can. Journal For Res.*, 23 (1993), 1980–1989.

[17] M. S. Peercy, M. Olano, J. Airey, and P. J. Ungar, Interactive multi-pass programmable shading, *SIGGRAPH 00 Conference Proceedings*, 2000.

[18] E. Ribbens, J. A. SIlander, and S. W. Pacala, Seedling recruitment in forests: calibrating models to predict patterns of tree seedling dispersal, *Ecology*, 75 (1994), 1794–1806.

[19] H. Samet, *Applications of Spatial Data Structures: Computer Graphics, Ima ge Processing, and GIS*, Addison-Wesley, Reading, MA, 1990.

[20] W. H. Schlesinger, *Biogeochemistry*, Morgan Kaufmanni Pub. Co., 1993.

[21] Shugart and West, Development of an appalachian deciduous forest succession model, *J. Environ. Manag.*, 5 (1977), 161–179.

[22] D. L. Urban, G. Bonan, T. Smith, and H. Shugart, Spatial applications of gap models, *Forest Ecol. Manage.*, 42 (1991), 95–110.