# Distributed Land-Cover Change Simulation Using PVM and MPI

M.W. Berry & K.S. Minser

Computer Science Department

CS-97-356 February 1997

# Distributed Land-Cover Change Simulation Using PVM and MPI

M.W. Berry and K.S. Minser

*Department of Computer Science, University of Tennessee, 107 Ayres Hall, Knoxville, TN, 37996-1301,* [berry,minser]@cs.utk.edu

**Abstract**

Computer simulations are used in landscape ecology to simulate the effects of human land-use decisions on the environment. Such decisions are influenced by both ecological and socioeconomic factors which can be represented by spatially explicit multidisciplinary data. The **L**and-**U**se **C**hange **A**nalysis **S**ystem (or **LUCAS**) was developed to study the effects of land-use on landscape structure in such areas as the Little Tennessee River Basin in western North Carolina and the Olympic Peninsula of Washington state. These effects include land-cover change and species habitat suitability. Using a geographic information system (GIS) to store, display and analyze map layers derived from remotely sensed images, census and ownership maps, topological maps, and output from econometric models, a parallel/distributed version of LUCAS (pLUCAS) was developed for simulations on a network of workstations. Targeting distributed computational environments reflects the resources available to most land-use planners, forestry personnel, and wildlife managers. A performance evaluation of two pLUCAS distributed models on an ATM-based network of 12 SUN Ultra-2 workstations is presented. Particular emphasis is given to the range of speed improvements (relative to serial runs on a single SUN Ultra-2 workstation) that can be obtained using the PVM or MPI message-passing environments.

*Keywords:* ATM-based network of workstations, distributed computing environment, ecological and socioeconomic factors, land-cover change simulation, MPI, parallel implementations, PVM.

⋆ This research has been supported by the Southeastern Appalachian Man and the Biosphere (SAMAB) Program under U.S. State Department Grant No. 1753-000574 and University of Washington Subcontract No. 392654, by the National Science Foundation under grants NSF-ASC-94-11394 and NSF-CDA-95-29459, and the USDA Forest Service under Contract No. 29-1286.

## 1 Introduction

Humans can have a direct influence on changes in the natural environment. One approach toward a better understanding of the the effects of human land-use decisions on the environment is to consider both ecological and socioeconomic factors. Such a multidisciplinary approach was taken by the Man and the Biosphere (MAB) project [2], whose goal was to analyze the environmental consequences of alternative land-use management scenarios in two different geographic regions: the Little Tennessee River Basin (LTRB) in North Carolina and the Olympic Peninsula in Washington State.

The MAB approach involved the integration of disciplines such as ecology, economics, sociology, and computer science to evaluate the impacts of land-use. This integration also required that data from the various disciplines share a compatible representation. Such forms include tabular and spatial databases, results of mathematical models, spatial models and expert opinions [2,7]. A *geographic information system* or GIS, such as the Geographic Resources Analysis Support System (GRASS) [12], can be used to easily store and manipulate the spatially explicit representation of this data. The Land-Use Change Analysis System (LUCAS) is a prototype computer application specifically designed to integrate the multidisciplinary data stored in GRASS and to simulate the land-use policies prescribed by the integration model.

### *1.1 Sample Scenario and Validation*

In LUCAS, scenarios describe prescribed land-use policies to be simulated. As an example, suppose that a natural resource manager in the LTRB would like to determine the impact of not logging any trees for 50 years on the habitat of the Wood Thrush (*Hylocichla mustelina*). The scenario is formally defined to use the historical transition probabilities based on existing map layers from 1975, 1980 and 1986 along with the restriction that once a grid cell of land is forested, it will remain forested. For example, the land manager may run LUCAS with 10 replicates for 10 time steps each to simulate the change over 50 years. The manager can examine the graphical statistics plotted on the screen or more carefully analyze the statistics saved to a SAS [8] file. Other scenarios with different constraints can be investigated and their results compared. In this way, the investigator can better understand the effects of potential land-use decisions.

To validate the LUCAS model, historical data are compared against the simulated data [2,7]. Starting with the oldest existing map, the period of time up to the year for which the newest map exists must be simulated. The degree to which the statistics for the simulated and historical land cover layers agree determines the accuracy of the model for this period.

### *1.2 LUCAS/pLUCAS Development*

The initial LUCAS prototype was implemented as an *object-oriented* C++ application to promote modularity. This modularity facilitates the addition of future software which might address the needs of different types of users. Future expansions of LUCAS are discussed in [2] and [7] while Section 2 describes (in brief detail) the modular implementation of the initial LUCAS prototype followed by the more recent parallel and distributed versions in Section 3. The creation of a distributed version [6] of LUCAS, Parallel LUCAS (pLUCAS), was motivated by the computational needs of real-time processing and extensions to larger regions. The first design of pLUCAS [6] utilized the Parallel Virtual Machine (PVM) [5] message-passing environment, and the (current) follow-up implementations are based on MPI [9]. The performance of both PVM and MPI implementations when tested on an ATM-connected network of workstations will be discussed in Section 4.

## 2 LUCAS Design

As discussed in [2,7], LUCAS provides a stochastic model for the future assessment of landscape change using historical maps of land cover. The initial design's modularity provides great flexibility for future modifications required by diverse users.

### *2.1 Stochastic Modeling*

The econometric model used in LUCAS is a dynamic, *stochastic* model primarily based on one random variable, namely land cover, and deals explicitly with time-variable interaction. The stochastic simulations enabled by LUCAS employ the statistical sampling of multiple replicates, i.e., repeated simulations of the same model. The statistical output produced by LUCAS is composed of SAS-compatible [8] data which can be imported by any generic graphing tool/software. Figure 1 outlines the modular model used to develop the LUCAS prototype. Each module of the LUCAS model is briefly described in the following sections (see [2] for more details).

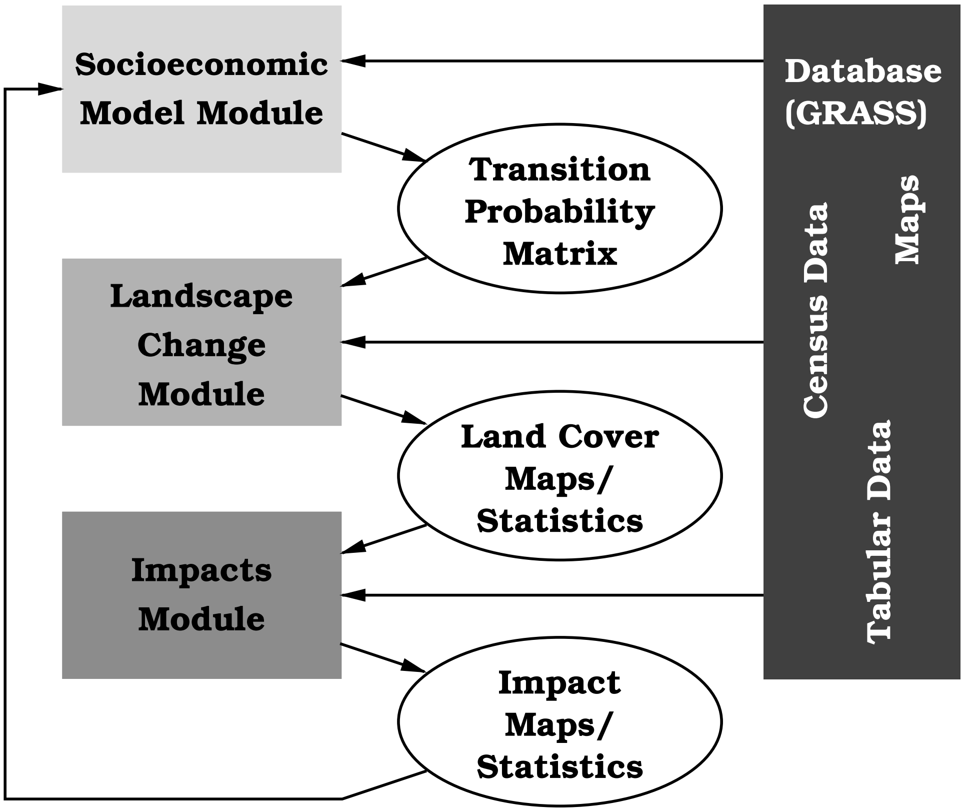



Fig. 1. Relationship among LUCAS modules

### *2.2 Socioeconomic Model Module and TPM*

Several discrete and continuous ecological and sociological variables were used empirically in calculating the probability of change in land cover: land-cover type (vegetation), slope, aspect, elevation, land ownership, population density, distance to nearest road, distance to nearest economic market center (town), and the age of trees. For an analysis of the influence of these economic and environmental factors on landscape change see [11]. Each variable corresponds to a spatially explicit map layer stored in the GIS. A vector of all of these values for a given grid cell is called the *landscape condition label* [3,4]. An example landscape condition label (LCL) [7] is shown in Table 1.

Each element of the LCL $\vec{x} = (x_1, x_2, \ldots, x_8)^T$ is used to determine the probability of change using the multinomial logit equation [14,13,2]

$$\Pr(i \to j) = \frac{\exp(\alpha_{i,j} + \vec{z}^T \vec{\beta}_j)}{1 + \sum_{k \neq i}^{n} \exp(\alpha_{i,k} + \vec{z}^T \vec{\beta}_k)}, \tag{1}$$

where $n$ is the number of cover types, $\vec{z}$ is a $5 \times 1$ column vector composed

Table 1
Example Landscape Condition Label in the Hoh Watershed on the Olympic Peninsula

| $\ell$ | $x_\ell$ | **Definition** | **Attribute** |
|---|---|---|---|
| 1 | 2 | Public Lands | Land ownership |
| 2 | 75 | 75 years old | Tree age (Olympic Peninsula only) |
| 3 | 1 | Coniferous | Land cover (vegetation) |
| 4 | 20 | 20° incline | Slope |
| 5 | 1 | True | Steep slope (> 17° incline; Olympics only) |
| 6 | 1500 | 1500 meters | Elevation |
| 7 | 1000 | 1000 meters | Distance along roads to nearest market center |
| 8 | 21 | 1890 meters | Distance to nearest road |
| 9 | 2 | 0.002 people/acre | Population density (LTRB only) |

of elements $x_4, \ldots, x_8$ of the LCL $\vec{x}$ in Table 1, $\vec{\beta}_j = (\beta_{1,j}, \beta_{2,j}, \ldots, \beta_{5,j})^T$ is a vector of logit coefficients, $\alpha_{i,j}$ is a scalar intercept, and $\Pr(i \rightarrow j)$ is the probability of coniferous land cover remaining the same ($i = x_3 = 1 = j$) at time $t + 1$ or changing to another cover class (i.e., $j = 2, 3, 4$). The land ownership ($x_1$) determines which table of logit coefficients should be used and the tree age ($x_2$), used only for coniferous forest cover types, determines if the trees have aged sufficiently to be harvested, i.e., change to another cover type. The *null-transition* or probability of no land cover change is defined by Equation 2.

$$\Pr(i \rightarrow i) = \frac{1}{1 + \sum_{k \neq i}^{n} \exp(\alpha_{i,k} + \vec{z}^T \vec{\beta}_k)}, \tag{2}$$

where the symbols have the same meaning as in Equation (1). Example vectors of coefficients for the Hoh and LTRB Watersheds are available in [2] and [7]. Such coefficients and associated intercept values have been calculated empirically by Wear *et al.* [14] from existing historical data stored in the GRASS database. The table of all probabilities generated by applying Equation (1) to all cover types is called the *transition probability matrix* (TPM), an example of which can be found in Table 2. If the TPM in Table 2 were used, for example, a random number from the closed interval $[0, 1]$ less than 0.8725 would signify that the land cover would remain coniferous. For a discussion of logistic regression and a basis for Equation (1) see [10].

Table 2
Example Transition Probability Matrix based on the example multinomial logit coefficients.

| **From Coniferous** | **Changing to** | **Probability** |
|---|---|---|
| $1 \rightarrow 1$ | Coniferous | 0.8725 |
| $1 \rightarrow 2$ | Deciduous/Mixed | 0.1186 |
| $1 \rightarrow 3$ | Grassy/Brushy | $1.886 \times 10^{-3}$ |
| $1 \rightarrow 4$ | Unvegetated | $6.989 \times 10^{-3}$ |

### *2.3 Landscape Change Module*

The Landscape Change Module in Figure 1 is the heart of the LUCAS software. On input, this module accepts the multinomial logit coefficients generated in Socioeconomic Model Module, implements the actual landscape change, and produces new land cover maps and statistics as output. The first step in designing LUCAS was to develop the method to simulate one time step, a five year period, of landscape change over multiple replicates.

Two types of transitions are simulated by LUCAS: grid cell (or pixel-based) and patch-based. The determination of the pixel-based landscape transitions is relatively trivial because each grid cell changes independently. The transition probabilities from the initial cover type to all other cover types are calculated using Equation (1) and the value of the landscape condition label of a grid cell. A pseudorandom number is then drawn from a uniform distribution between 0 and 1. This number, in turn, determines the new land cover type for this grid cell via the calculated probabilities. Patch-based transitions are considerably more difficult because of the task of patch identification. A patch (or cluster) is a group of contiguous grid cells with identical landscape condition labels. Although patch identification was not used in this research effort, algorithms for determining both the number and structure of patches (clusters) is available [2].

#### *2.3.1 Statistics*

Once the map of new land cover has been generated, the ecologist or land manager can use the results to determine the impact of the policy defined in the Socioeconomic Model Module. As stated in Section 2.1, statistics are the only true metric for analyzing a stochastic simulation. They also provide a convenient method for understanding the impact of the particular land management policy or scenario. The statistics in Table 3 are collected by LUCAS for each time step.

Table 3
Statistics collected by LUCAS

| | Statistic |
|---|---|
| **Pixel Statistics** | Proportion of landscape in each cover type<br>Area (ha) of landscape in each cover type<br>Edge:area ratio for each cover type<br>Amount of edge (km) for each pair of cover types<br>Total edge (km) in the whole landscape |
| **Patch Statistics** | Number of patches<br>Mean patch size<br>Standard deviation of patch size<br>Size of largest patch<br>Cumulative frequency distribution of number of patches by size<br>Mean patch shape (normalized shape index) [1] |

### *2.4 Impacts Module*

The land cover maps produced by the Landscape Change Module (see Section 2.3) are analyzed by the Impacts Module. This module may eventually determine the effect the changed landscape has on species, habitats, water quality, or other environmental impacts. Currently LUCAS is designed to perform only species' habitat suitability analyses [2,7]. Although an extensive list of species and habitat identification algorithms for each of the watersheds currently simulated are available, this module was not used in the results presented in Section 4. The usual output from this particular module is a binary map; either a grid cell is suitable for a species or it is not. The statistics in Table 3 are again collected for each impact map.

## 3 pLUCAS Implementations Using PVM and MPI

The parallel and distributed implementation of LUCAS (pLUCAS) is based on the same functional design of the serial prototype described in Section 2. The motivation for pLUCAS was to manage the multiple independent replicates required (for accuracy purposes) in the stochastic simulation of land-cover change. As most end users of LUCAS would not be expected to have multiprocessor computing systems available, software that could exploit a network of workstations was considered more desirable.

Parallelization is used so that each processor performs one complete simulation

[1] This is the corrected patch perimeter/area metric: $(0.282 \times \text{perimeter}/\sqrt{\text{area}})$ [1].

(replicate) of LUCAS exclusive of any other processor or process. The statistics calculated from each replicate are stored locally to disk (on each processor) until all replications are completed. At that time, the statistics are assembled on the main node and stored for later use. In the performance tests presented in Section 4, 10 replicates of 20 timesteps each are performed for 4 different scenarios on the Hoh Watershed of the Olympic Peninsula. Along with these 40 tasks (4 scenarios $\times$ 10 replications) is a task for the initialization of any future impact modules that could be used (see Section 2.4) so that a total of 41 independent tasks are scheduled.

The initial version of pLUCAS was implemented using PVM [6] and has now been modified to use the MPI message-passing software library [9]. All versions emulate the basic host/worker model (described below) with some differences inherent to PVM and MPI. All pLUCAS runs using PVM and MPI were tested on a network of twelve Sun Microsystems Ultra 2 workstations, each containing two 167-Mhz UltraSPARC-1 processors under the Solaris 2.5.1 operating system. Each workstation had 256-Mbytes of memory and two 2.1-Gbyte internal disks. Peak performance of one UltraSPARC-1 processor is about 126 Mflops (millions of floating-point operations per second). The workstations were connected by both a 10 Mbps Ethernet interface and 155 Mbps ATM sbus adapter so that performance results (recorded in elapsed wall-clock time) could be obtained with two different network latencies.

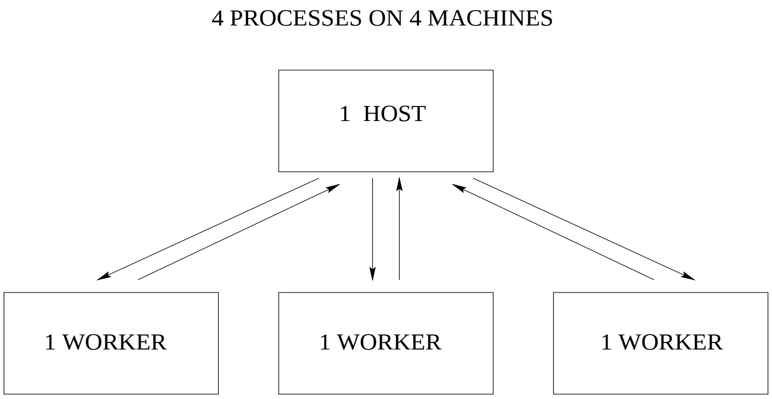
### *3.1 PVM Version*

The PVM (version 3.3.10) implementation of pLUCAS allows the host process to assign tasks to worker processors by spawning a worker process onto a specific machine. The host maintains a queue of tasks to be scheduled, tasks completed, and available workers (machines). After all tasks are completed, the host spawns new processes on all machines to send data accumulated from their previous tasks back to the machine owning the host process. The host process collects the data, assembles it, and writes the results to files stored on a machine external to the ATM-connected network. The host processor is allowed to spawn a worker process to itself. Thus, the host machine will have one host and one worker process assigned to it. All other machines will have only one worker process at a time.

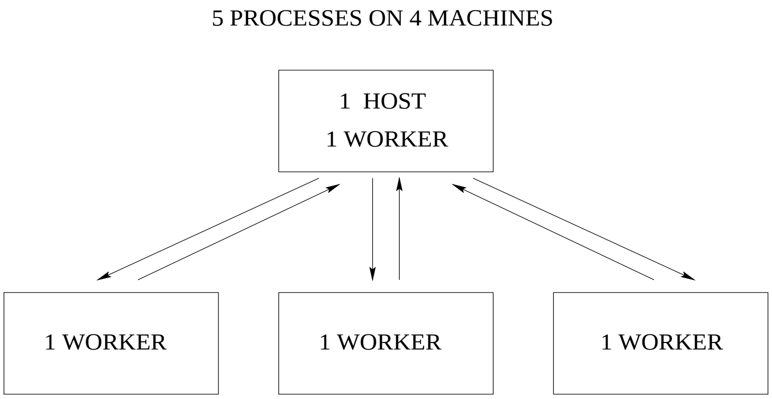
### *3.2 MPI Versions*

To incorporate MPI (version 1.0.13) into the pLUCAS software, major code revisions were necessary due to the lack of process spawning with MPI. A traditional *host/worker* model was implemented using a top-level `if-then-else` construct to select the appropriate set of instructions for each process type. All tasks/duties assigned to the workers and the final accumulation of statistics from all processes (see Section 3.1) are accomplished via message-passing.

Two different versions of MPI have been developed. The first method does not allow a worker process to coexist on the same machine as the host process. Therefore, a 4-machine network has only 3 worker processes as shown in Figure 2(a). This setup is referred to as MPI(1) in all subsequent tables and graphs and is defined as k processors on k machines. The second MPI version is a better emulation of the PVM approach (see Section 3.1) which allows for 1 worker process to be assigned to the host process machine. Therefore, a 4-machine network would have one host process and 4 worker processes. This method is referred to as MPI(2) on all subsequent tables and graphs and defined as $k$ processes on $k$-1 machines (see Figure 2(b)).




(a) MPI(1): $k$ processes on $k$ machines




(b) MPI(2): $k$ processes on $k$-1 machines

Fig. 2. Two MPI-based implementations of the distributed pLUCAS model.

## 4 Performance Results for pLUCAS

In order to test the speed and scalability of the pLUCAS implementations described above, several experiments were conducted on the network of Sun Ultra 2 workstations described in Section 3. These experiments involved 3 runs each with 1, 2, 4, 8, and 12 workstations to compute 10 replicates of 20 time steps for each of the four historical, pixel-based scenarios of the Hoh Watershed on the Olympic Peninsula shown in Table 4. Figure 3 illustrates one of the Hoh land-cover maps obtained before and after a 100-year simulation (using Scenario 1 from Table 4).

Results reported for a single Sun Ultra 2 workstation reflect the use of the serial LUCAS implementation (i.e., no message-passing overhead). The elapsed wall-clock times recorded for the experiments reported in this section are provided in Tables 5 and 6 in the Appendix. These wall-clock times do not include the installation of GIS data (done only once before any experiments were run) on a local disk of each machine on the ATM-connected network.

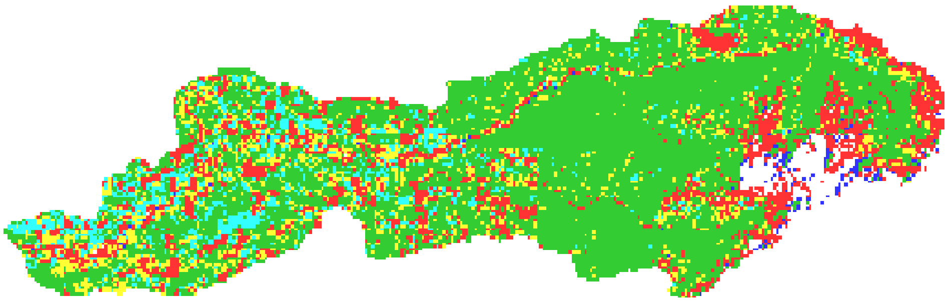
Table 4
Scenarios of land-cover change for Hoh Watershed according to historical transition probabilities

| | **Ownership Type** | |
|---|---|---|
| **Scenario** | **Public** | **Private** |
| 1 | 1986–1991 | 1986–1991 |
| 2 | 1986–1991 | 1975–1986 |
| 3 | 1975–1986 | 1986–1991 |
| 4 | 1975–1986 | 1975–1986 |

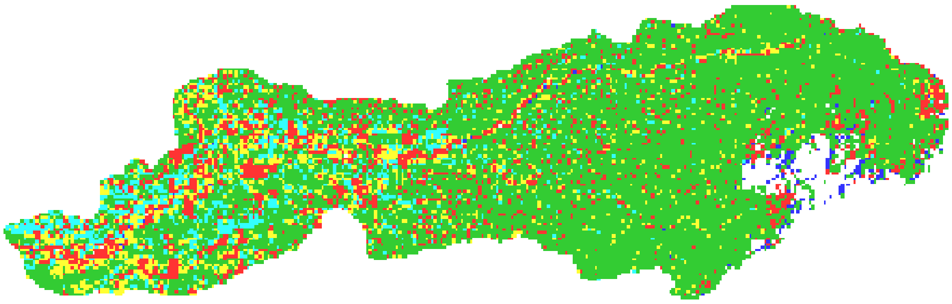
### *4.1 PVM Versus MPI*

For both PVM and MPI, the ATM (155 Mbps) network was slightly faster than the Ethernet (10 Mbps) network, but the difference was not significant (see Figures 4 and 5). The fact that the only message-passing in pLUCAS occurs between the host and the workers (none between workers) might account for the small improvement of ATM over Ethernet. PVM had the best performance over either method of MPI with MPI(2) performing most similar to PVM. However, as more machines are added to the network, MPI(2) began losing its advantage over MPI(1) (see Figure 6).

Serial time for LUCAS on a single machine was 28.27 minutes. With 12 machines, the PVM implementation completed in 3.61 minutes yielding a speedup of 7.83. For the same number of machines, MPI(1) finished in 3.86 minutes followed by MPI(2) in 3.87 minutes with speedups of 7.32 and 7.30, respectively.



(a) Before simulation of Scenario 1



(b) After simulation of Scenario 1

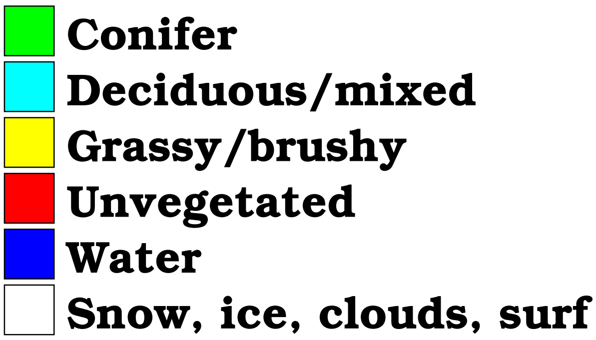



Fig. 3. Hoh Watershed maps before and after a 100-year simulation

With a 4-node machine, MPI(2) out-performed MPI(1) with speedups of 3.29 and 2.68, respectively. For an 8-node machine, MPI(2) yielded a speedup of 5.54, but MPI(1) maintained a speedup of 5.47. Finally, increasing the machine to 12 nodes resulted in MPI(1) and MPI(2) showing similar performances with MPI(1) actually being slightly faster than MPI(2).

### *4.2 MPI(1) Versus MPI(2)*

The faster deteriorating performance of MPI(2) compared to that of MPI(1) can be attributed to process contention. Recall that for the MPI(1) scheme, the host process is scheduled on a dedicated processor and does not compete with any worker process. As more machines are included in the network, more message-passing demands are required of the host. In the MPI(2) scheme, the host machine has a host process as well as a worker process competing for the same port [2] for message-passing. As the number of messages increase with added machines, the network contention becomes aggravated for MPI(2).

In order to validate the network contention suffered by MPI(2), the idle time or *wait-time* for message-passing was measured on a 12-node machine. The elapsed wall-clock time for waiting was measured using the function **MPI_WTime()** which returns the actual wall-clock time in seconds. Two calls to **MPI_WTime()** were made (before and after each blocking send and receive). The time returned from the first call is subtracted from the second call to yield the elapsed time of the message-passing function. The wait-times of the message-passing routines were summed for an entire run on each processor. The accumulated wait-time incurred is divided by the number of tasks assigned to that particular processor to determine the average wait-time per task on each processor (workstation). In some circumstances, not all processors performed the exact same number of tasks within a complete experiment. These averages were determined from 6 runs of both MPI(1) and MPI(2). The range of wait-times as well as the mean are illustrated in Figure 7 and listed in Table 6 of the Appendix.

For both MPI(1) and MPI(2), processor 3 contained the host process, and in the case of MPI(2), processor 3 contained both the host and a worker process. Although the wait-time of processor 3 for MPI(2) was not significantly high, the wait-times for MPI(2) across all 12 processors was much higher than those obtained with MPI(1) inferring a more congested network for MPI(2). Adding more processors yielded greater wait-times, and hence the time improvements for the MPI(2) quickly deteriorated.

[2] Although each Sun Ultra 2 workstation had two 167-Mhz UltraSPARC-1 processors, contention for the single ATM port degraded message-passing latencies when more than 1 PVM or MPI process was scheduled on a given machine.

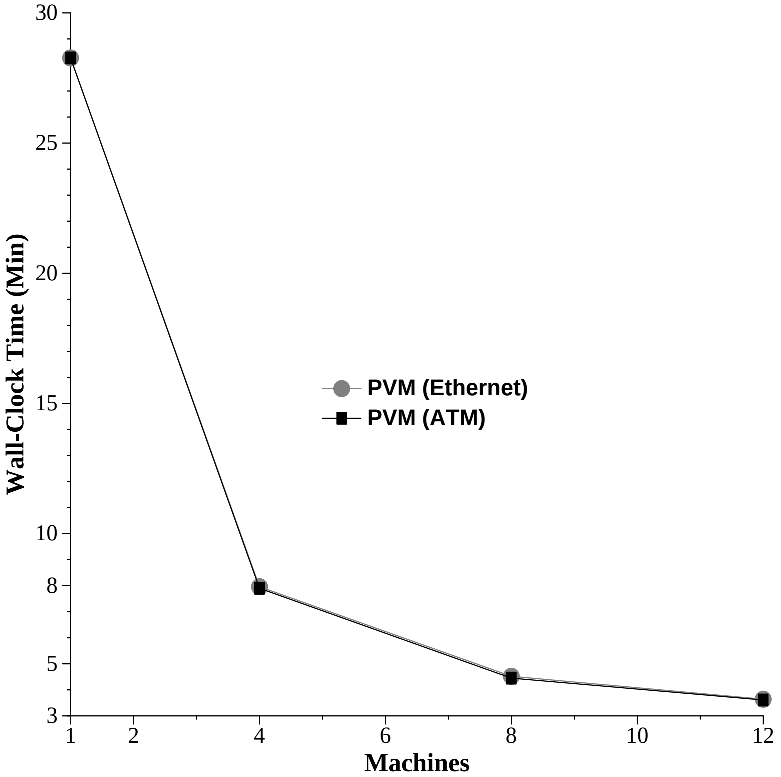
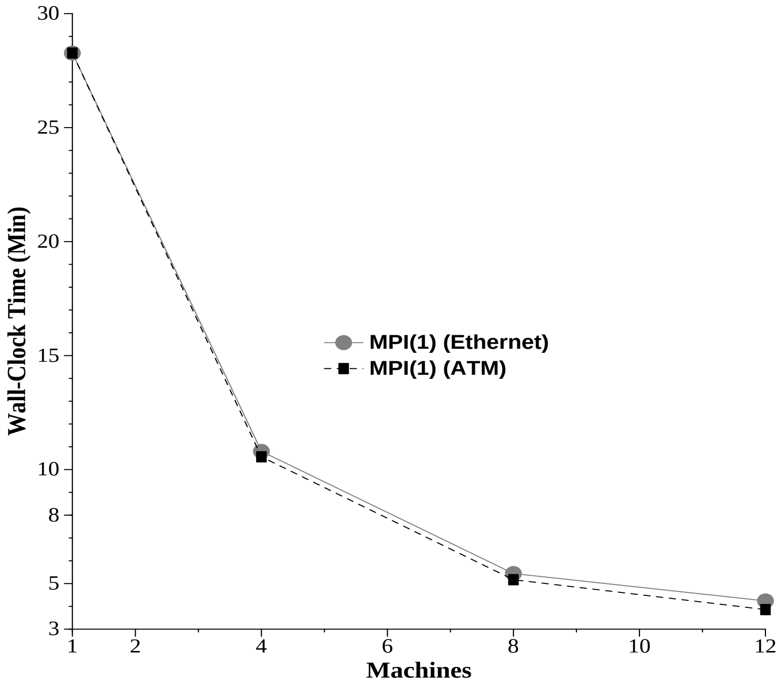
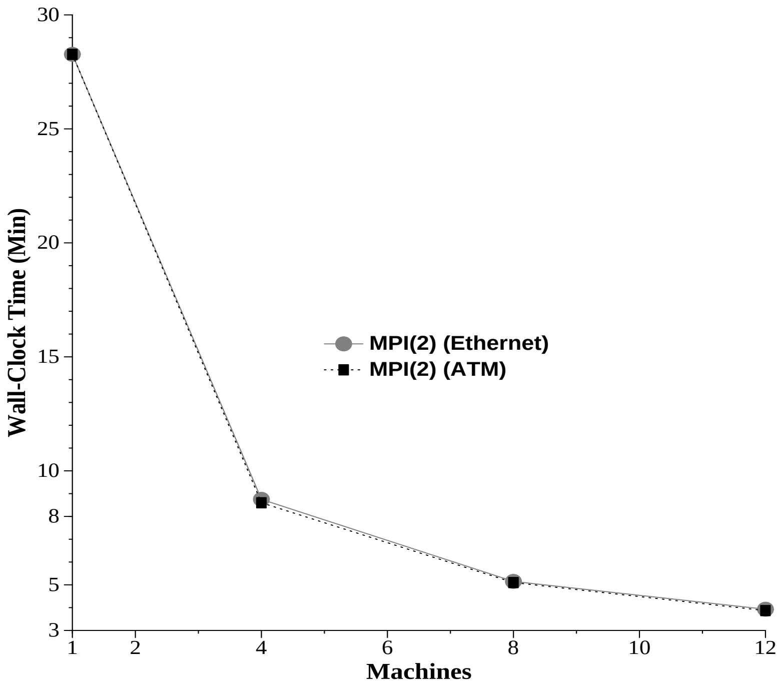



Fig. 4. Timing comparisons for PVM-based implementation on ATM and Ethernet.

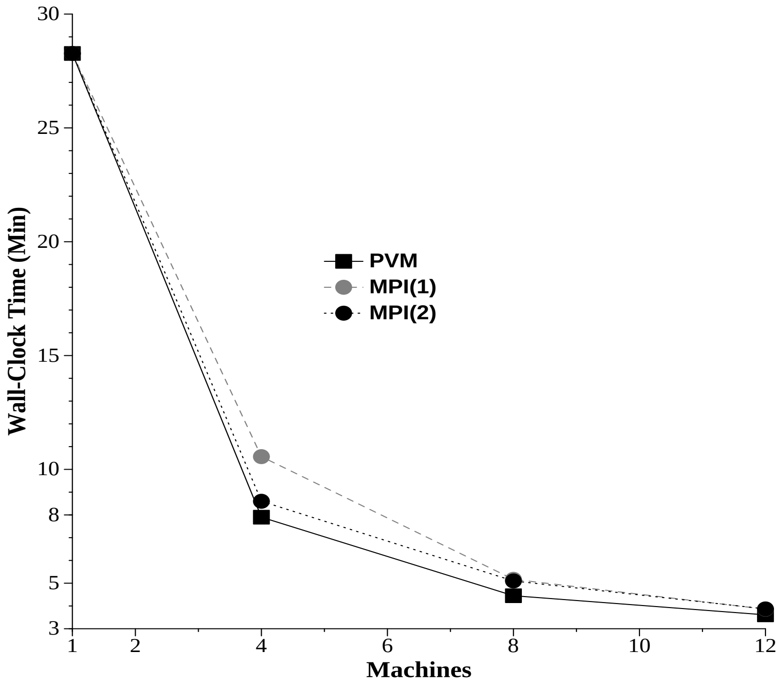



(a) MPI(1) performance ($k$ processes on $k$ machines)

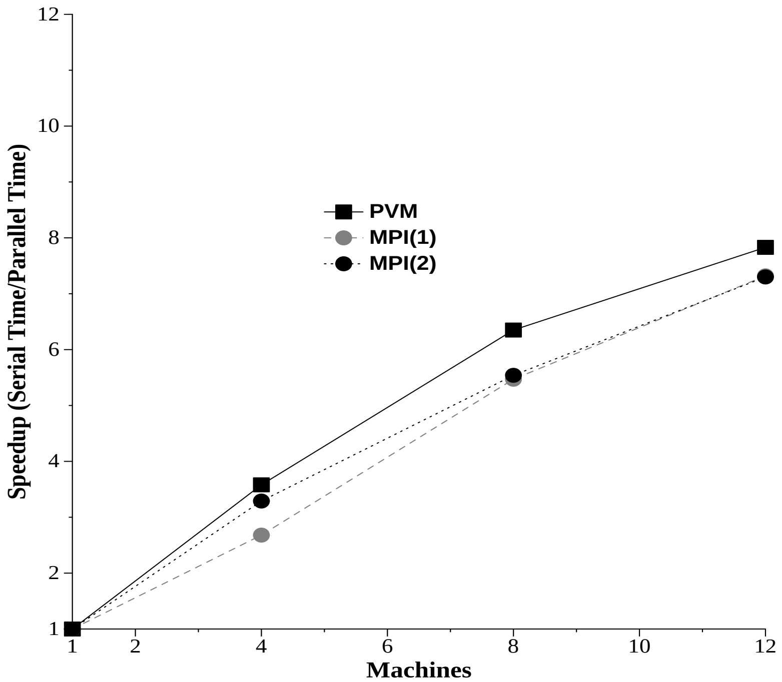



(b) MPI(2) performance ($k$ processes on $k$-1 machines)

Fig. 5. Timing comparisons for MPI(1,2) implementations of pLUCAS using ATM and Ethernet connections.

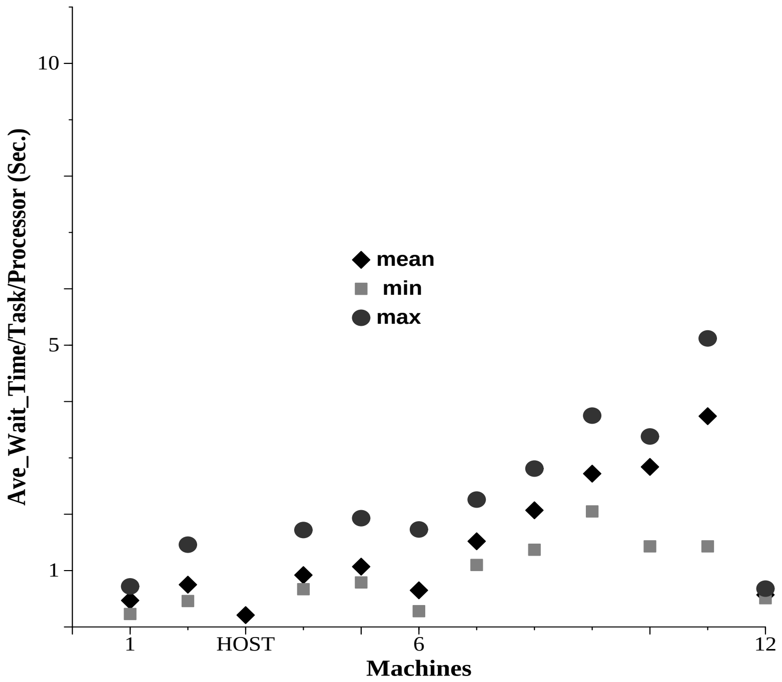



(a) Timing comparison

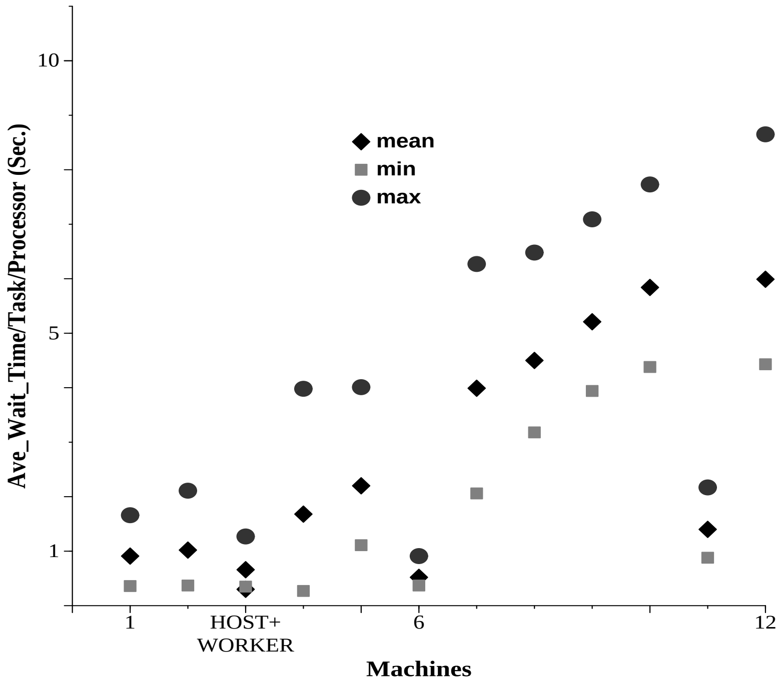



(b) Speedup comparison

Fig. 6. Timing and speedup comparisons of all three pLUCAS versions (PVM, MPI(1), and MPI(2)) using an ATM-connected network.




(a) MPI(1) implementation ($k$ processes on $k$ machines)




(b) MPI(2) implementation ($k$ processes on $k$-1 machines)

Fig. 7. Average wait-times per process for each machine when using the MPI(1,2) implementations of pLUCAS on an ATM-connected network.

## 5 Conclusions and Future Work

The Land-Use Change Analysis System (LUCAS) is a valuable problem solving environment for modeling landscape changes. pLUCAS offers a distributed solution to computational demands of stochastic simulation on a network of workstations. No significant differences were observed in the performance of ATM versus Ethernet with the PVM and MPI implementations of pLUCAS. Although the PVM implementation did produce faster execution times for all numbers of machines on the network, the MPI(2) implementation did perform equally well. As the network size grew, the network congestion suffered by MPI(2) offset the potential speedup gain with more machines. The host/worker distributed model used in MPI(1) was certainly less sensitive to aggravated network congestion since the host process did not share machine resources with any worker process.

Future software development of the pLUCAS prototype includes the porting of the MPI implementations to a recently acquired IBM SP-2 multiprocessor system (having 40 computational nodes), and a more thorough investigation of how two or more processes scheduled on one of the Sun Ultra 2 workstations (having two physical processors) can better time-share a single ATM port. Future modeling work with LUCAS and pLUCAS includes the integration of multidisciplinary data from several forestry growth and production models (funded by the Environmental Impacts Program of the USDA Forest Service).

## References


[1] W. L. Baker and Y. Cai. The r.le programs for multiscale analysis of landscape structure using the GRASS geographical information system. *Landscape Ecology*, 7(4):291–302, 1992.

[2] M. W. Berry, R. O. Flamm, B. C. Hazen, and R. L. MacIntyre. Lucas: A System for Modeling Land-Use Change. *IEEE Computational Science and Engineering*, 3(1):24–35, June 1996.

[3] R. O. Flamm and M. G. Turner. Alternative model formulations for a stochastic simulation of landscape change. *Landscape Ecology*, 9(1):37–46, 1994.

[4] R. O. Flamm and M. G. Turner. GIS applications perspective: Multidisciplinary modeling and GIS for landscape management. In V. Alaric Sample, editor, *Forest Ecosystem Management at the Landscape Level: The Role of Remote Sensing and Integrated GIS in Resource Management Planning, Analysis and Decision Making*, pages 201–212. Island Press, Washington, D.C., 1994.

[5] A. Geist et al. *PVM: Parallel Virtual Machine A User's Guide and Tutorial for Networked Parallel Computing.* MIT Press, Cambridge, MA, 1994.

[6] B. C. Hazen. A Distributed Implementation of the Land-Use Change Analysis System (LUCAS) Using PVM. Master's thesis, University of Tennessee, Knoxville, August 1995.

[7] B. C. Hazen and M. W. Berry. The Simulation of Land-Cover Change Using a Distributed Computing Environment. *Simulation Practice and Theory*, 1997. In Press.

[8] SAS Institute Inc., Cary, NC. *SAS User's Guide: Statistics*, 1992.

[9] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI: The Complete Reference*. The MIT Press, Cambridge, MA, first edition, 1996.

[10] J. C. Trexler and J. Travis. Nontraditional regression analyses. *Ecology*, 74(6):1629–1637, 1993.

[11] M. G. Turner, D. N. Wear, and R. O. Flamm. Influence of Land Ownership on Land-Cover Change in the Southern Appalachian Highlands and the Olympic Peninsula. *Ecological Applications*, 1996. In Press.

[12] U.S. Army Construction Engineering Research Laboratory, Champaign, IL. *GRASS 4.1 User's Reference Manual*, Spring 1993.

[13] D. N. Wear and R. O. Flamm. Public and private forest disturbance regimes in the southern Appalachians. *Natural Resource Modeling*, 7(4):379–397, Fall 1993.

[14] D. N. Wear, M. G. Turner, and R. O. Flamm. Ecosystem management in a multi-ownership setting: Exploring landscape dynamics in a southern Appalachian watershed. *Ecological Applications*, 1995. In Press.

## Appendix

Table 5
Wall-clock times (in minutes) and speedups for all three pLUCAS implementations using Ethernet- and ATM-connected networks.

(a) Wall-clock times in minutes

| | PVM | | MPI(1) | | MPI(2) | |
|---|---|---|---|---|---|---|
| Machines | Ethernet | ATM | Ethernet | ATM | Ethernet | ATM |
| 4 | 7.96 | 7.90 | 10.80 | 10.56 | 8.75 | 8.60 |
| 8 | 4.52 | 4.45 | 5.44 | 5.17 | 5.15 | 5.10 |
| 12 | 3.64 | 3.61 | 4.24 | 3.86 | 3.93 | 3.87 |

(b) Speedups

| | PVM | | MPI(1) | | MPI(2) | |
|---|---|---|---|---|---|---|
| Machines | Ethernet | ATM | Ethernet | ATM | Ethernet | ATM |
| 4 | 3.55 | 3.58 | 2.62 | 2.68 | 3.23 | 3.29 |
| 8 | 6.25 | 6.35 | 5.20 | 5.47 | 5.49 | 5.54 |
| 12 | 7.77 | 7.83 | 6.67 | 7.32 | 7.19 | 7.30 |

Table 6
Wait-time per task per machine for MPI(1,2) on each machine using an ATM-connected network.

(a) MPI(1)

| nprocs | 1 | 2 | 3 (H) | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| min | 0.23 | 0.46 | 0.21 | 0.67 | 0.79 | 0.28 | 1.10 | 1.37 | 2.05 | 1.43 | 1.43 | 0.51 |
| mean | 0.47 | 0.75 | 0.21 | 0.92 | 1.07 | 0.65 | 1.52 | 2.07 | 2.72 | 2.84 | 3.74 | 0.57 |
| max | 0.72 | 1.46 | 0.21 | 1.72 | 1.93 | 1.73 | 2.26 | 2.81 | 3.75 | 3.38 | 5.12 | 0.68 |

(b) MPI(2)

| nprocs | 1 | 2 | 3 (H) | 3 (W) | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| min | 0.36 | 0.37 | 0.24 | 0.35 | 0.27 | 1.11 | 0.37 | 2.06 | 3.18 | 3.94 | 4.38 | 0.88 | 4.43 |
| mean | 0.91 | 1.02 | 0.30 | 0.66 | 1.68 | 2.20 | 0.52 | 3.99 | 4.50 | 5.21 | 5.84 | 1.40 | 5.99 |
| max | 1.66 | 2.11 | 0.36 | 1.27 | 3.98 | 4.01 | 0.91 | 6.27 | 6.48 | 7.09 | 7.73 | 2.17 | 8.65 |