To the Graduate Council:

I am submitting herewith a thesis written by Ember Uziel entitled "Parallel Animal Migration Models of Bison and Elk in Northern Yellowstone National Park." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Michael W. Berry, Major Professor

We have read this thesis
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor
and Dean of the Graduate School

Parallel Animal Migration Models of Bison and Elk in Northern Yellowstone National Park

A Thesis

Presented for the

Master of Science Degree

The University of Tennessee, Knoxville

Ember Uziel

December 1994

Acknowledgments

I thank my thesis advisor, Dr. Michael Berry, for his endless encouragement, guidance, and patience. I also thank Dr. Micah Beck and Dr. David Straight who served on my thesis committee, and the University of Tennessee Computer Science Department for the opportunity to complete a graduate degree. I thank Dr. Monica Turner for the opportunity to work with her Northern Yellowstone Park model and for her expert advice. I thank my husband, Jeff Constantin, and my parents, Mayo and Mary Uziel, for their endless support and proof-reading skills. I also thank James Stevens, system administrator, and Randy Bond, lab manager, for maintaining the CM5 and its users, which was quite demanding. Lastly, I thank Jane Comiskey, who paved the way for this thesis and provided invaluable insight and information.

Abstract

In landscape ecology, computer models are developed to simulate the migration of animal groups as they exist in nature. Typically, these models are sequential and the animal groups move and forage in sequence. In this work, we discuss the parallelization of the NOYELP (NOrthern YELlowstone national Park) model on a 32-processor Thinking Machines CM-5. NOYELP is a spatially-explicit individual-based model that simulates the search, movement and foraging activities of groups of animals across the northern portion of Yellowstone National Park. NOYELP is primarily used to study the effects of fire scale and pattern on the winter foraging dynamics and survival of free-ranging bison and elk groups. Among four versions of a parallelized NOYELP model (referred to as PNOYELP), the version (AM) based on arithmetic means of shared variables is shown to provide daily energy gain statistics closest to the sequential NOYELP model with reasonable speed improvements (ranging from 2.8 to 6.7) over the sequential model executing on a Sun SPARCstation 10.

Contents

1 Introduction **1**

1.1 Overview 1

1.2 Motivation 2

1.3 Parallel Computing Environment 2

2 Northern Yellowstone National Park Model **4**

2.1 Study Area 4

2.2 Model Description 5

2.2.1 Landscape Representation 6

2.2.2 Forage Distribution 6

2.2.3 Snow Simulation 7

2.2.4 Ungulate Initial Conditions 7

2.2.5 Ungulate Foraging 8

2.2.6 Search and Movement Rules 9

2.2.7 Ungulate Energetics 11

2.2.8 Program Flow 11

3 Parallel Model **14**

3.1 Division of Sequential Code 14
3.2 Partitioning of the Landscape 16
3.3 Node Initialization 18
3.4 Animal Movement 20
3.4.1 Determination of New Processor 20
3.4.2 PMI Average 21
3.4.3 Moving to New Processor 22

4 Verification and Performance **25**
4.1 Comparison of Selected Outputs 25
4.1.1 Arithmetic Mean 26
4.1.2 Harmonic Mean 36
4.1.3 Arithmetic Mean with Percentage Threshold 36
4.1.4 Partial Averages 37
4.2 Performance of Parallel Versions 38
4.3 Performance of PNs 40
4.4 Recommendations 42

5 Summary and Future Work **43**

Bibliograpy **47**

Appendices **50**

A NOYELP Specifics **51**
A.1 Outline of NOYELP Subroutines 52
A.2 Selected Formulas used in NOYELP model computations 58

B PNOYELP Subroutines 61

B.1 Outline of PNOYELP Subroutines 62

C Daily Energy Balance Statistics 65

Vita 67

List of Tables

3.1 Percentage of time spent in each function of the original sequential and modified sequential versions of NOYELP. 15

4.1 Wall-clock times (in seconds) for the NOYELP and PNOYELP models with a varying number of elk. Speed improvements of each parallel version over the original sequential version are also listed. 39

List of Figures

2.1 NOYELP landscape with 285 row × 584 column bounding box. 6

2.2 Search and movement patterns of animal groups in NOYELP. 10

2.3 NOYELP model program flow by subroutine. 13

3.1 PNOYELP program flow by subroutine. 16

3.2 NOYELP landscape with a regular grid partitioning scheme. 17

3.3 NOYELP landscape with an irregular grid partitioning scheme. 17

3.4 CPU time (in seconds) of the original sequential NOYELP model based on different animal totals. 18

3.5 Partition A with nearest-neighbor partitions (B). 19

3.6 Animal group A determines a new processor to move to and is placed within its maximum moving distance on the new processor (C). . . . 23

4.1 Daily travel distance of elk cows for 5 repetitions. 27

4.2 Daily survival of elk cows for 5 repetitions. 27

4.3 Sequential model ungulate location map for day 1. 29

4.4 Parallel model ungulate location map for day 1. 29

4.5 Sequential model ungulate location map for day 120. 30

4.6 Parallel model ungulate location map for day 120. 30

4.7 Sequential model ungulate location map for day 155. 31
4.8 Parallel model ungulate location map for day 155. 31
4.9 Sequential model ungulate location map for day 135. 32
4.10 Sequential model ungulate location map for day 140. 32
4.11 Sequential model ungulate location map for day 145. 33
4.12 Sequential model ungulate location map for day 150. 33
4.13 Parallel model ungulate location map for day 135. 34
4.14 Parallel model ungulate location map for day 140. 34
4.15 Parallel model ungulate location map for day 145. 35
4.16 Parallel model ungulate location map for day 150. 35
4.17 Partial averages compared by an animal group located on processor A. 38
4.18 Total computation time on PNs for 5 repetitions of the AM version. . 41
4.19 Total number of animals processed by each PN for 5 repetitions of the AM version. 41

Chapter 1

Introduction

1.1 Overview

Computer models can be used to study the responses of ungulates to environmental heterogeneity [TWW+93]. Models from landscape ecology, in particular, attempt to simulate interactions between animals and vegetation. The NOYELP (northern Yellowstone National Park) landscape ecology model used in this thesis was originally developed by Drs. Monica Turner and Yegang Wu of the Environmental Sciences Division, Oak Ridge National Laboratory. NOYELP is a spatially explicit individual-based simulation model developed to explore the effects of fire scale pattern on the winter foraging dynamics and survival of free-ranging elk (*Cervus elaphus*) and bison (*Bison bison*) in northern Yellowstone National Park. The model is used to project ungulate survival on the winter range under different fire regimes and winter weather scenarios [TWW+93].

1.2 Motivation

The primary goal of this thesis was to reduce the effective run time of the NOYELP simulation via parallel processing. Because of the stochastic nature of NOYELP, it required multiple simulations that often demanded several hours of run time in the past on RISC-based workstations. Until now, no other related landscape ecology models have been implemented on a multiprocessor architecture; thus, another motivation for parallelizing the NOYELP model was to examine the effect of allowing the animal groups to move in parallel as opposed to the sequence dictated by the sequential model. In order to parallelize the original model (NOYELP), assumptions (i.e., determining animal movement, determining animal moving order, and determining animal search area) were made that were not in the sequential model.

1.3 Parallel Computing Environment

The MIMD-architecture used in this thesis for parallelizing the NOYELP model was a 32-processor Thinking Machines CM5. The 32 processing nodes (PNs) which are scheduled by a single control processor (CP) (also known as the partition manager or PM) collectively form 1 partition of a *fat-tree* data network [Hwa93]. In general, the size of the partition can range from tens to thousands of processors. Each partition runs the CMOST operating system, an enhanced version of UNIX. With the vector unit accelerators (VUs) installed, each PN has four VUs that provide enhanced performance for parallel programs that are arithmetic in nature. Using the VUs, each PN is capable of achieving 128 megaflop/s (millions of floating-point operations per second) so that the peak performance of the 32-processor CM5 is 4 gigaflop/s (billions of floating-point operations per second). For this thesis, however, the vector units

were not used since a majority of the required computation is integer-based.

Access to the PNs in a partition is accomplished through the CP. Programming on the CM5 utilizes one of two paradigms: *host/node* and *hostless*. In the *host/node* paradigm, one processor, usually the CP, acts as the *host* for the program and controls both the program flow and the workload assigned to each PN. Also in the *host/node* paradigm, the program is not limited to executing the same program on each of the PNs. It is feasible to execute different programs working on independent data on each of the PNs in the partition (MIMD). In the *hostless* paradigm, all of the PNs are executing the same program, on the same data or different data, in parallel without the influence of a CP. The CMOST operating system provides a limited *host* program to control message passing and I/O service request generated by the PNs. This thesis used the *host/node* paradigm in order to allow the animals to move independently on each PN.

Communication between the CP and the PNs or between individual PNs is accomplished through message-passing routines that are supplied by CMMD version 3.2. The FORTRAN-77 compiler (`f77`), supplemented with synchronous message-passing library routines, was used on each CM5 node. Both point-to-point and broadcast message-passing paradigms were used in order to simulate animal movement and track daily energy balance statistics.

Chapter 2

Northern Yellowstone National Park Model

(NOYELP) is a spatially explicit stochastic simulation model developed to explore the effects of fire scale and pattern on the winter foraging dynamics and survival of free-ranging elk (*Cervus elaphus*) and bison (*Bison bison*). The search, movement, and foraging activities of individuals or small groups of ungulates are simulated in the model. Following [TWW+93], this chapter summarizes the NOYELP model. A summary of formulas, used in search, movement and forage calculations, and subroutines from the sequential NOYELP model can be found in Appendix A.

2.1 Study Area

The Yellowstone National Park (YNP) was established in 1872 and encompasses 9000 km^2 in the northwest corner of Wyoming and adjacent parts of Montana and Idaho. The NOYELP model focuses on the northern 20% of the Park, which covers

77,020 hectares. The climate is characterized by long, cold winters and short, cool summers [Dia79, DM82]. About 83% of the elk's winter range, which is warmer and drier than the rest of the Park, is included within YNP. The vegetation consists of continuous forests at higher elevations and sagebrush-grasslands interspersed with coniferous forests at lower elevations. During 1988, approximately 34% of the winter range burned [DRSS89]. Simulations presented in this thesis use the fire pattern data attained from the 1988 fires.

2.2 Model Description

NOYELP is a spatially explicit model that simulates the search, movement and foraging activities of small groups of bison and elk. The landscape is influenced by different fire disturbances and winter severities which affect the foraging activities. Forage intake is a function of a group's initial body weight, the quantity of forage available on a site, and the depth and density of snow. The group's endogenous reserves are reduced to offset deficits when the energy expenditures of a group exceed the energy gained during a day.

The landscape is represented as a gridded, irregular polygon with a spatial resolution of 1 hectare. A time step of 1 day is used in the simulation for a total of 180 days spanning November 1 through the end of April. Because the model does not project ungulate reproduction or plant regrowth during spring and summer, simulations are conducted anew for each winter season.

Figure 2.1: NOYELP landscape with 285 row × 584 column bounding box.

2.2.1 Landscape Representation

Because of the irregular shape of the northern range, a 285 row by 584 column pixel map (or grid cell map) encompasses the 77,020 grid cells (Figure 2.1). Spatial heterogeneity across the landscape is represented by several data layers (i.e., fire pattern, slope, aspect, habitat, and elevation) obtained from the YNP geographic information system (GIS).

2.2.2 Forage Distribution

An initial quantity of available forage is assigned to each grid cell based on its habitat type and burn status at the beginning of each simulation [BCM94]. As the ungulate groups graze the forage on a given pixel, the absolute abundance of forage on that pixel is decremented. Since the model does not simulate regrowth or reproduction, forage abundance either remains constant or declines over the 180 day simulation.

2.2.3 Snow Simulation

Snow conditions are an extremely important determinant of winter ungulate dynamics, thus, the model must simulate the spatial and temporal heterogeneity of these conditions across the northern range. The northern range is subdivided into two subregions for the snow simulation. One subregion is in a rain shadow, thus it has low precipitation levels, while the rest of the landscape comprises the second snow subregion.

Snow simulation is a two-step process. In the first step, baseline snow depth and snow density, which are used to determine energetic cost of travel and the maximum daily moving distance, are projected within each of the two subregions. Monthly data for snow depth, snow-water equivalent, and weather are used for the baseline snow projections. Snow-water equivalent, an integrated measure of snow mass, is used to influence daily forage intake of the animals. The change in snow conditions from month to month is assumed to be linear. A 3-day interval for updating snow conditions is used in the simulations for this thesis, but the model allows the user to specify any time interval. The second step involves the distribution of snow to each grid cell. This process is accomplished by modifying the baseline projection based on the slope and aspect of each grid cell.

2.2.4 Ungulate Initial Conditions

Six ungulate classes are represented in the model: cows, calves, and bulls for both elk and bison. The herds of elk and bison are initialized with a total number of ungulates. Each herd is then divided into calf, cow, and bull groups that will remain together for the entire simulation. All ungulate groups are assigned an initial bodyweight based

on their ungulate class. Since the animal groups move one by one, they are assigned a randomly generated number (*animal group number*) to determine their moving order. Each calf group is assigned to a cow group and moves accordingly to the same site.

Equations for updating snow and search, movement and foraging activities (described in Appendix A) apply to individual animals; the group size is used as a multiplier. Elk cows, calves, and bulls maintain a group size of 4 animals. Bull cows and calves have a group size of 9 and bison bulls travel in a group size of 2.

Initially, elk are placed randomly across the entire landscape on grid cells containing forage. Bison are also distributed randomly on resource sites, but only in nonforest habitats at elevations less than 2100 meters. More than one group of elk or bison may simultaneously occupy a single grid cell throughout the simulation.

2.2.5 Ungulate Foraging

If there is forage above a threshold (explained in Chapter 3) available on a grid cell, an ungulate group located there will graze. The group will continue to graze throughout the day until it reaches its maximum daily intake, which is a function of the initial body weight of the ungulates at the beginning of winter.

Two negative feedback terms, ranging from 0 to 1, are included in the model to regulate ungulate foraging. A feedback value of 1, indicating no limitation, allows the group the maximum forage intake, while a feedback value of 0 allows no forage intake. The lower of the two feedback terms is allowed to operate on the daily forage intake.

The first feedback term, `ynpr`, represents the effect of a reduction in the amount of available forage on the rate of forage intake. As animals graze at a grid cell, this feedback value is decreased to represent a smaller amount of forage. The biomass at

a grid cell must be above a certain level, the refugium level, otherwise the ungulates will not be able to graze.

The second feedback term, `fbsm`, represents the effect of snow on the ability of an animal to obtain forage. Because animals generally cannot forage in snow that exceeds their brisket height, snow depth may limit foraging. Snow density can also be a problem since shallow dense snow may preclude foraging. Thus, snow-water equivalent, which integrates both depth and density, is used as a feedback term.

2.2.6 Search and Movement Rules

The algorithm used to simulate search and movement for the animal groups is a simple rule-based algorithm. If a group is located on a grid cell with available forage, the group will graze. Otherwise, the group will search in concentric rings, up to a radius equal to the maximum moving distance, for another grid cell with available forage. Maximum moving distance is a function of ungulate type and snow conditions. Available forage is determined by the worse of two feedback terms. One is based on absolute abundance of forage (`ynpr`) and the other (`fbsm`) is based on snow conditions on the grid cell. The feedback term with the lower value, the one that will determine movement, is stored in the PMI variable (preferential moving index). The ungulate group will move to a grid cell, within a concentric ring, with the highest feedback term greater than 0.1. The threshold value of 0.1 was empirically derived. If multiple grid cells in a concentric ring have the same feedback value, the ungulate will randomly choose one of the grid cells with the highest value. The same movement sequence is used if the animal group is located on a pixel without available forage. If no available forage is found within the search radius, the animal group is placed randomly at its maximum moving distance. For example, (Figure 2.2) illustrates the search pattern

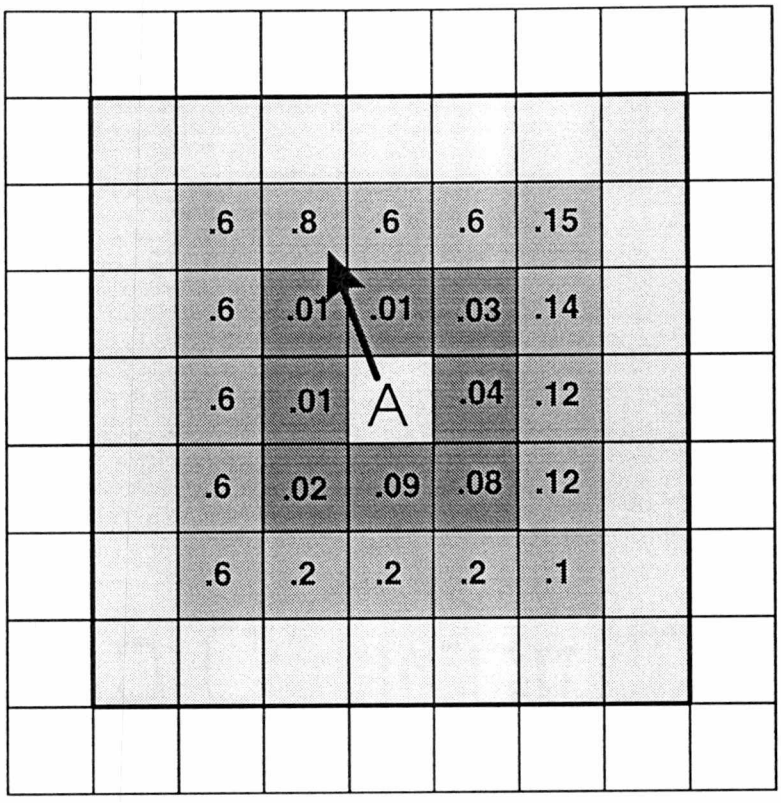

Figure 2.2: Search and movement patterns of animal groups in NOYELP.

of an animal group with a maximum moving distance of 3 pixels, and the selection of a new grid cell. The animal group will search the dark gray grid cells first. The number in a grid cell represents a PMI value at that pixel. Because none of the PMI values in the first concentric ring are above the threshold of 0.1, the group continues searching in the next concentric ring (medium gray). Many grid cells in this ring have a PMI value above 0.1, thus the animal group selects the grid cell with a PMI value of 0.8, which is the highest value in that concentric ring. If the group had not found a suitable grid cell in this concentric ring, it would have continued the search until it had reached its maximum moving distance (black border).

The ungulate group will continue to search each day until it has met one of two

conditions: it has reached its maximum moving distance, or it has reached its maximum daily intake. If the group has reached its maximum moving distance for the day, the group is not allowed to forage any more that day. The next ungulate group is then located on the landscape and the procedure repeated until all ungulate groups have completed foraging for the day.

2.2.7 Ungulate Energetics

Daily energy balance, the difference between energy gain and cost, are computed at the end of each day for each ungulate group. Energy gain is a function of the total amount of forage consumed each day, while energy cost is dependent on maintenance energy and energy required for travel. Maintenance energy includes the cost of all the animal's daily activities: standing, resting, grazing, ruminating, and thermo regulating. The cost of travel is based on actual distance traveled and snow conditions at the grid cell the animal is leaving.

After the daily energy balance is computed, the body weight of the ungulates is computed. If the energy balance is positive, the ungulates do not gain weight, otherwise they lose weight.

2.2.8 Program Flow

The major subroutines defining the program flow of the NOYELP model (see Appendix A) are illustrated in Figure 2.3. Static data layers and ungulate specifics are the inputs to subroutine `input`, while dynamic data layers and ungulate initial conditions are initialized in `initial`. `ssnow` updates snow conditions every 3 days. `ungfind`, `graze`, `move`, and `energet` comprise the subroutines needed for the movement sequence.

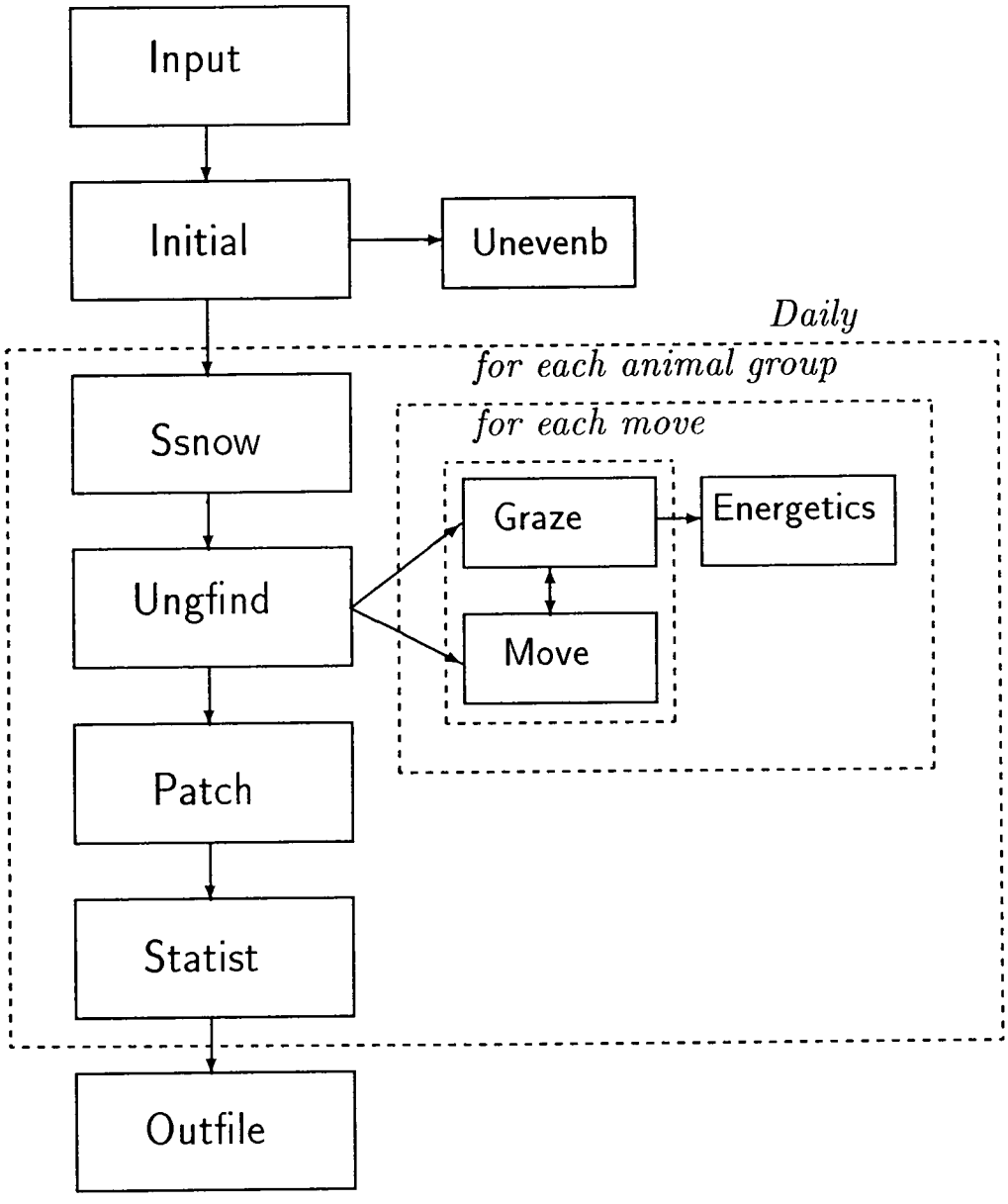

Figure 2.3: NOYELP model program flow by subroutine.

ungfind locates an animal group on the landscape and allows it to begin the movement sequence discussed earlier in this chapter. graze allows the animal to forage on a grid cell and energet records the daily energy balance statistics. patch is an optional subroutine that identifies clusters of high and low biomass patches. The simulations presented in this thesis did not use the patch subroutine. outfile is another optional subroutine which will write dynamic data layers (i.e., available biomass, ungulate location) to file.

Chapter 3

Parallel Model

The parallel model requires changes to the original sequential model in order to optimize the code for the 32-processor Thinking Machines CM5. This chapter discusses the necessary modifications and the justification for the changes as well as new assumptions used to create the parallel model. The original sequential subroutines are detailed in Appendix A and the subroutines modified for the parallel model are provided in Appendix B.

3.1 Division of Sequential Code

In order to parallelize the sequential NOYELP program, the code was divided into two portions: one portion that would run on the host and another portion that would run in parallel on the 32 PNs of the CM5 (described in Chapter 1). In order to determine where time was spent, the original sequential program was profiled using the UNIX intrinsic function `gprof` on a Sun SPARCStation 10. Profiling revealed that functions `main`, `move`, and `ungfind` comprised the bulk of time in NOYELP

Table 3.1: Percentage of time spent in each function of the original sequential and modified sequential versions of NOYELP.

Function Name	Original Sequential	Modified Sequential
main	13.7	0.2
`input`	0.2	0.1
`initial`	0.1	0.2
ssnow	3.8	5.0
ungfind	14.0	29.2
graze	5.0	2.3
move	53.2	56.1
energet	1.0	0.8
`statist`	0.0	0.0
`outfile`	0.0	0.0
other	9.0	6.1
Percentage Parallelizable		93.3

Note: The italicized functions were chosen to run in the node portion of the parallel version.

(Table 3.1). Since the move sequence for the animal groups includes the `move` and `ungfind` functions, all functions involved in the move sequence were transferred to the node program of the parallel version. These transferred functions included `ungfind`, `graze`, `move`, and `energetics`. The `ssnow` function was also chosen to run on the nodes to avoid having the host update the nodes every 3 days with new snow conditions. Most of the `main` function was also implemented in the node portion of the program because it did not have to run sequentially. After the host and node sections were separated within the sequential program, it was profiled again with `gprof` and over 93% of the original sequential program could be run in parallel on the nodes (Table 3.1). From Amdahl's law [Hwa93], this implies that the maximum speedup possible on 32 processors of the CM5 is given by

$$S_{32} = \frac{1}{0.07 + \frac{.93}{32}} \simeq 10.1, \tag{3.1}$$

assuming a fixed work load. In order to attain a fixed work load in both the parallel

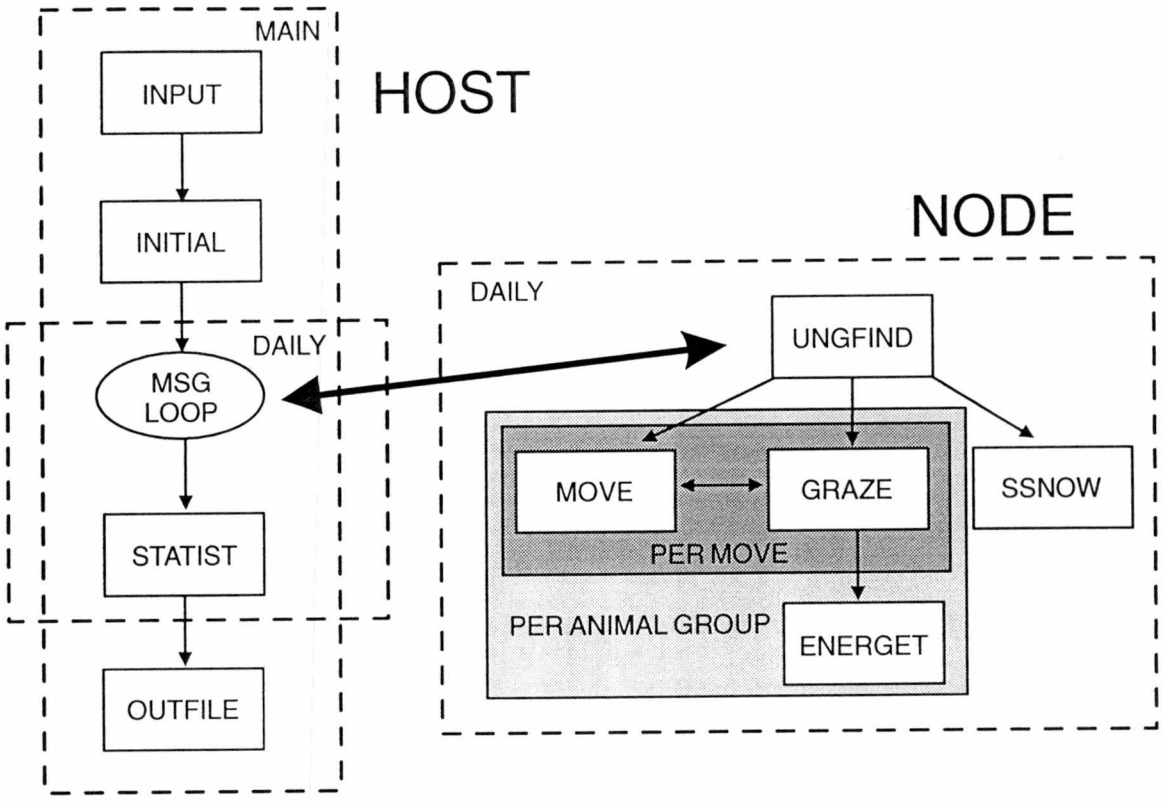

Figure 3.1: PNOYELP program flow by subroutine.

and sequential models, the parallel model would require the same survival rate as that of the sequential model. Figure 3.1 illustrates the program flow of the parallel model.

3.2 Partitioning of the Landscape

Once the NOYELP program was divided into a host portion and node portion, the landscape was partitioned across the 32 processors nodes of the CM5 in order to balance the work load on each processor. Initially, a regular grid was used to partition the landscape among the processors (Figure 3.2). Such a regular grid, however, forces

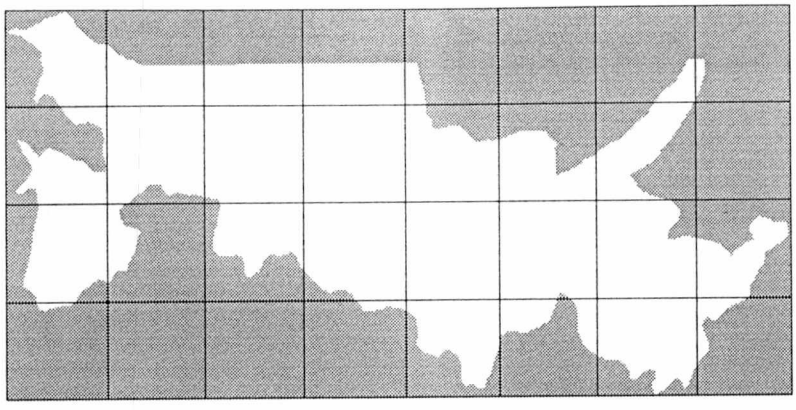

Figure 3.2: NOYELP landscape with a regular grid partitioning scheme.

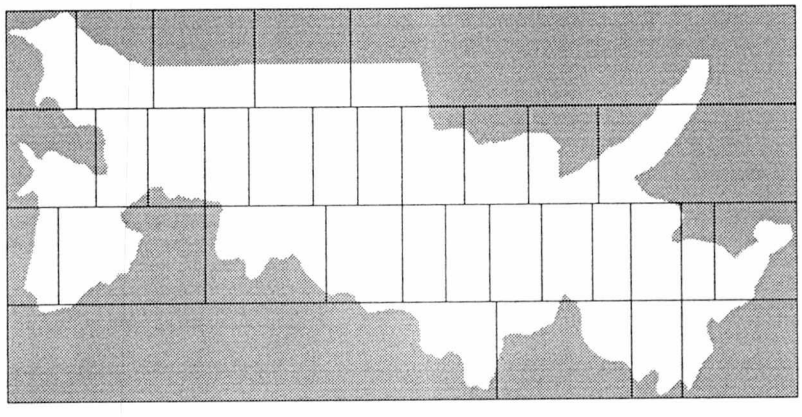

Figure 3.3: NOYELP landscape with an irregular grid partitioning scheme.

an obvious imbalance in the amount of work across the PNs as some processors are not allocated any portion of the study area.

The regular grid was then replaced with an irregular grid (Figure 3.3) for two reasons. One was to ensure a more balanced computational load across the PNs, and the second was to create a suitable partition on which the animal groups would find enough food and thus remain on the processor for as many days as possible (i.e., animal group localization). In order to derive a suitable irregular grid, the amount of available biomass on each partition was analyzed to determine how long the animal groups would remain on a given partition. The number of animals per processor was

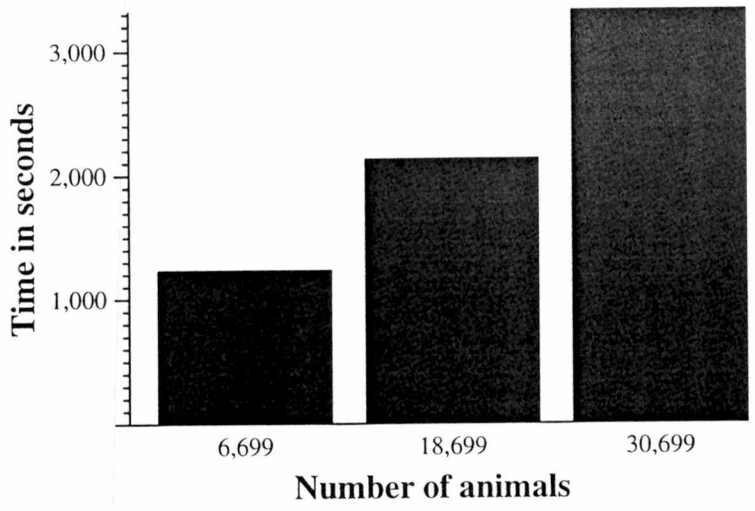

Figure 3.4: CPU time (in seconds) of the original sequential NOYELP model based on different animal totals.

also taken into account since time spent in the sequential model is very dependent on the number of animals (Figure 3.4).

Another design criteria for the irregular grid was potential dynamic resizing of the grid during the 180-day simulation. Without this consideration, a more equitable partitioned grid could have been created without row boundaries. Although no dynamic repartitioning of the landscape was actually used in the thesis, a strategy for refining the irregular grid in Figure 3.3 is discussed in Chapter 5.

3.3 Node Initialization

In the parallel implementation of NOYELP, the host program begins much like the sequential version. The data layers (elevation, habitat, aspect, slope, snow, and initial forage) are initialized along with ungulate initial conditions as discussed in Chapter 2. The host must then broadcast this information to the 32 PNs so each will be able to determine which animal groups and what portion of landscape they are

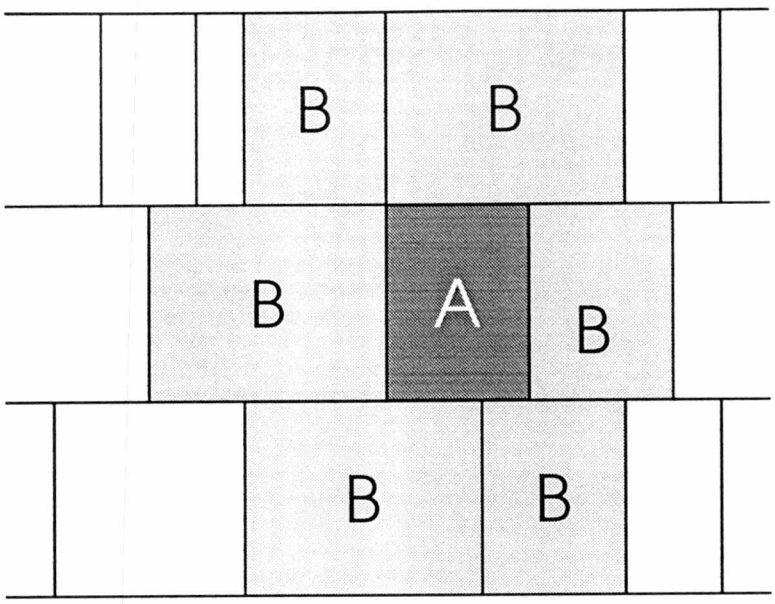

Figure 3.5: Partition A with nearest-neighbor partitions (B).

responsible for. The information is also used to update snow conditions and to allow the animal groups to move and graze over the landscape.

The node program then determines its nearest-neighbor partitions of the landscape based on the irregular grid. A nearest-neighbor is any partition that touches one of a partition's borders. Figure 3.5 illustrates processor A with its nearest neighbors (labeled with a B). This information is necessary so animal groups have access to neighboring partitions which they can possibly move to. Since the nearest-neighbor partitions are determined during the simulation based on the initial irregular grid, the grid could be repartitioned during the simulation and nearest neighbors recalculated (discussed in Chapter 5).

The last step in the node initialization requires the nodes to determine which animal groups it owns. This is done by placing a 1 in the array `nung_yes`, which has an entry for each group. This data structure was used to maintain the random

ordering of the animal groups while moving between processors. Each processor is responsible for recording and updating vital statistics (daily energy balance) for animal groups resident on its portion of the landscape and initiating their movement sequence.

3.4 Animal Movement

Once node initialization is complete, the animals are ready to begin the 180-day movement sequence. The movement sequence in the parallel model (PNOYELP) is very similar to that of the sequential model (NOYELP). The random moving order, determined by the *animal group number*, is preserved on each processor. The animal group with the lowest group number is located on each processor and allowed to start the move and graze sequence. The group will search the landscape on its processor within its maximum moving distance for a pixel with a PMI value greater than 0.1 (as discussed in Chapter 2). Unlike the sequential model where the group continues its move/graze sequence until it has met one of two conditions, the parallel model will discontinue the move/graze sequence if one of three conditions is met: it has reached its maximum daily intake, it has reached its maximum moving distance, or it cannot find a suitable grid cell within its maximum moving distance on its processor.

3.4.1 Determination of New Processor

If the animal group searches its processor within its maximum moving distance and cannot find a suitable pixel (one with a PMI value greater than 0.1), the animal group determines if there is nearest-neighbor processor it can move to. The new processor must be within its maximum moving distance and have a PMI average greater than

0.1 (discussed in the next two sections). If both conditions are satisfied, the animal group is allowed to move to the other processor to continue its movement sequence. Otherwise, the animal group is placed randomly at its maximum moving distance on its processor.

3.4.2 PMI Average

The PMI average, used to determine if an animal group is allowed to move to another processor, is calculated at the beginning of each day. The PMI value at each habitat pixel (a pixel in the study area) is summed and divided by the total number of habitat pixels to attain a PMI arithmetic mean for each partition. Other methods were also used to determine the PMI average and will be discussed in Chapter 4.

The average on each processor is then broadcast to every other processor, so each processor will have a copy of every other processor's PMI average. Even though each processor only needs to know the PMI averages of its nearest neighbors, a broadcast to all processors was used because of its superior performance over point-to-point communication on the CM5 [Cor93].

Using a PMI average can significantly reduce communication time [Com93]. Three data layers (`fbsm`, `ynpr`, and `biotot`) are needed to calculate the PMI value at each pixel. The `biotot` data layer gives the absolute amount of forage available at a given grid cell. The `fbsm` data layer is the feedback term based on snow, and `ynpr` is the feedback term based on available forage as discussed in Chapter 2. For the `ynpr` and `biotot` data layers, the value corresponding to the pixel the group has chosen to graze on will change, so that if these data layers were exchanged between nearest neighbors once a day, they would be invalid as soon as a group grazed on another pixel. Also, in order to exchange the data layers between nearest neighbors once a

day, 171 messages would need to be sent and received between nodes. In a worst case scenario, the node with the largest partition of the landscape would be required to send 2.5 million bytes of information in order to update its neighbors once per day.

3.4.3 Moving to New Processor

Once a group has decided to occupy a new processor's partition of the landscape, the movement sequence for that group is halted and variables needed to continue movement on the next processor are saved. The animal group number is saved so that the destination processor will know when to allow the animal to begin its move/graze sequence for the day. The current bodyweight is saved for the daily energy balance calculations at the end of the day. The current row and column number will be needed to determine placement of the animal group on the destination processor. The amount of food the group has grazed and the distance traveled so far that day is saved so that the animal group can continue its move/graze sequence until it reaches its maximum moving distance or its maximum daily intake. The snow density coefficient is saved to calculate maximum moving distance for the day and for the daily energy balance calculations. If a cow group has a calf group, all corresponding calf information is also saved for the destination processor. The sending processor also includes its processor number so the receiving processor will be able to correctly place the animal group on its landscape partition.

The group searching for a remote processor's partition to occupy will determine which nearest-neighbor PNs are within its maximum moving distance for that day. If one or more neighbor PNs satisfies that condition, the group will check the PMI average of each of those processors and move to the neighboring PN with the greatest PMI average. Figure 3.6 illustrates an example of a group choosing a new processor

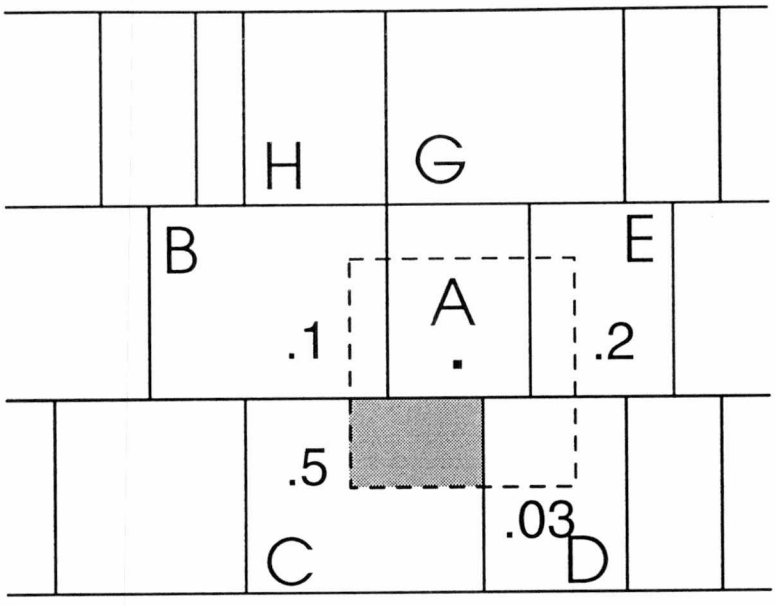

Figure 3.6: Animal group A determines a new processor to move to and is placed within its maximum moving distance on the new processor (C).

to move to. The group located on processor A has a maximum moving distance defined by the dashed line square surrounding it. Since the group's maximum moving distance falls within the landscape partition owned by processor's B, C, D, and E, the group must determine which of these nearest neighbors has the largest PMI average (the number in each partition). Processor C's PMI average is above the suitability threshold (0.1) and is larger than the other processors' PMI averages, thus the group will be placed randomly within its maximum moving distance on processor C (shaded grey).

If several neighboring PNs within the maximum moving distance have the same average, the group will choose the destination PN randomly among the set of neighboring PNs. The group is then removed from the processor's ungulate location map and the list of groups it owns. If a cow group has a calf group associated with it, the calf group is also deleted from the ungulate location map and from the list of groups

that processor owns.

The movement sequence for the remaining ungulate groups continues until the end of the day. Then, a message is sent to each nearest-neighbor PN containing either an array with ungulate information or a null message. The receiving processor accepts the message and checks for ungulate information or a null message. If there is group information, the message is unpacked and data layers and ungulate information are updated on the destination processor, so the animal group can continue its move/graze sequence. The animal group is then placed randomly within its maximum moving distance on the receiving processor and its move/graze sequence is continued for that day.

Chapter 4

Verification and Performance

In order to validate the parallel model, daily energy balance statistics and ungulate location maps from the parallel and sequential models were compared and analyzed. This chapter will discuss the differences and similarities between the selected outputs from the parallel and sequential models and will evaluate the effect of using different PMI averages. This chapter will also present performance results and speed improvements obtained with all versions of the parallel model.

4.1 Comparison of Selected Outputs

The daily energy balance statistics, listed in Appendix C, are used to record characteristics of the animal groups and landscape. Two of the daily energy balance statistics, *survival rate* and *daily travel distance* of elk cows, can be used to verify the entire model, and hence serve as the chosen statistics for comparison of the NOYELP and PNOYELP models. These statistics were plotted and compared to determine if

the animal groups were moving the same distance with a similar survival rate in both models.

4.1.1 Arithmetic Mean

The original parallel version (AM) used an arithmetic mean to compute the PMI average on each processor (Chapter 2). Because NOYELP is a stochastic model, daily energy balances are calculated over 5 repetitions, each with a different random number seed. In Figures 4.1 and 4.2, similarities in both the daily travel distance and the survival rate of elk cows in the NOYELP and the PNOYELP models are illustrated. A slight increase in ungulate deaths in the parallel model is explained by the greater distance traveled (as seen about day 165 through day 180 in Figure 4.1), which is due to the fact that the groups are forced to search their processor for suitable grazing pixels before they are allowed to move to another processor. Requiring an animal group to search its local partition before checking other processors may cause one or both of two events which can increase daily travel distance: (1) a group may find a suitable grid cell on its own processor, but at a further distance, or (2) a group might require more moves within its current processor's landscape partition in order to consume its maximum daily intake. Since the groups are moving further in the parallel model, they expend more energy and lose more weight, thus ungulate mortality increases. Even though there are differences between the models in the energy balance statistics, the differences are considerably small. The *daily survival rate* graph illustrates the range of 0% to 8% difference between the PNOYELP and NOYELP versions, while 0% to 25% is the range of differences in the *daily travel distance* graph until day 165. After day 165, the discrepancy grows as discussed before.

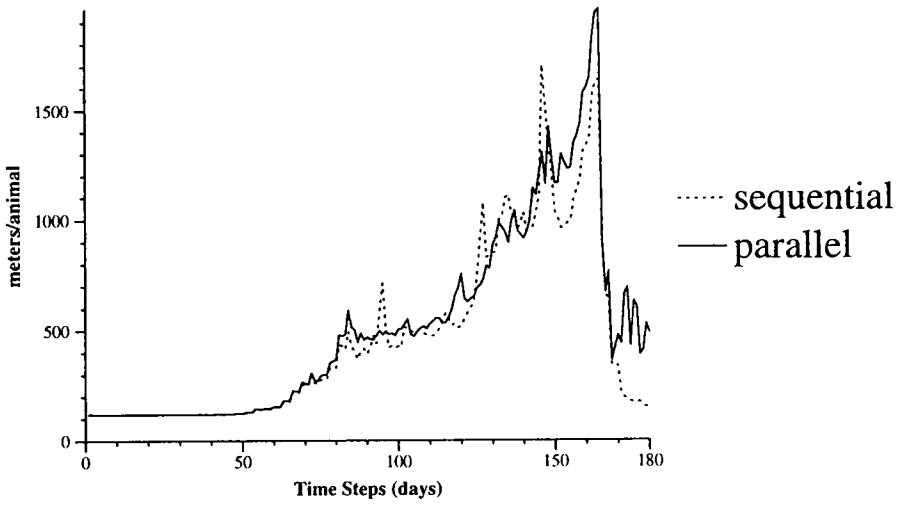

Figure 4.1: Daily travel distance of elk cows for 5 repetitions.

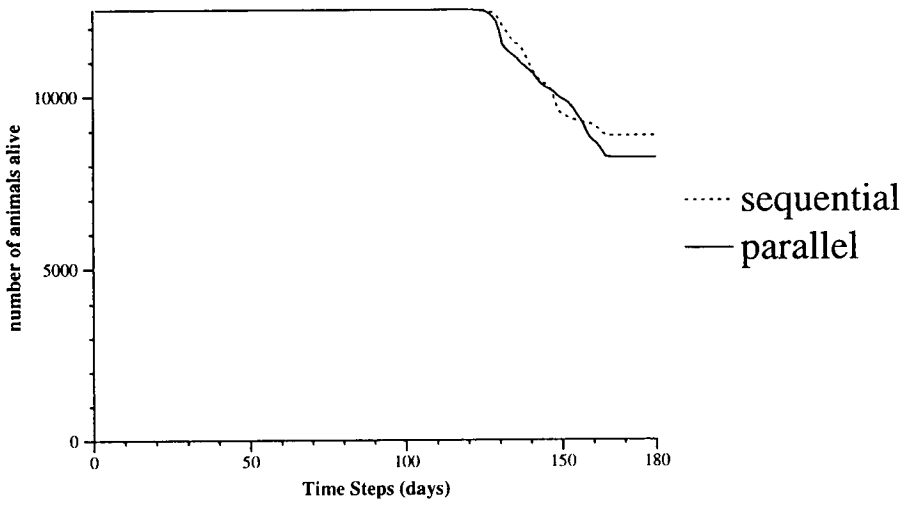

Figure 4.2: Daily survival of elk cows for 5 repetitions.

The ungulate location maps, which show the location of ungulate groups across the NOYELP landscape, are also used to validate the parallel model. On day 1, the ungulate groups are placed randomly across the landscape (Chapter 2), as illustrated in Figures 4.3 and 4.4. In these figures, each dot on the landscape represents between 1 and 5 groups per grid cell.

By day 120, the animal groups in both the sequential and parallel models have moved in the same direction and tend to converge to the same areas of the landscape (Figures 4.5 and 4.6). By day 155 (Figures 4.7 and 4.8), there are two noticeable differences between the parallel and sequential ungulate location maps: (1) processor partition definition is visible in the parallel map because the groups search within their maximum moving distance on their local processor before attempting to move to a new processor, and (2) a large cluster of groups of animals has congregated in the northwestern portion of the landscape partition (assigned to PN 1). The density of animal groups on PN 1 is a result of the animal groups in PNOYELP moving to the same landscape partitions a few days later than their counterparts would in the sequential model. To verify that the groups in PNOYELP were traveling in the same direction as the groups in NOYELP, ungulate location maps were dumped every 5 days between day 135 and day 150 and the maps indicate that some aspects of the group's locations and movement patterns are well represented in the parallel model (Figure 4.9 through 4.16). With either the sequential or parallel versions, the groups in both models move in a clumped-fashion along the northern border towards the west and then move south along the western border.

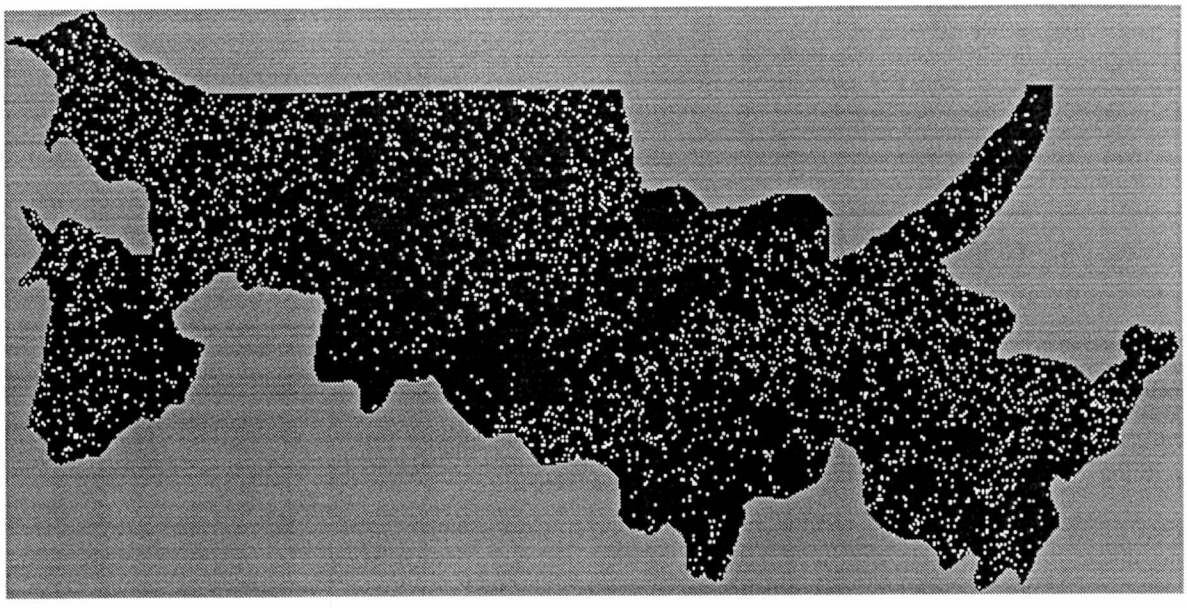

Figure 4.3: Sequential model ungulate location map for day 1.

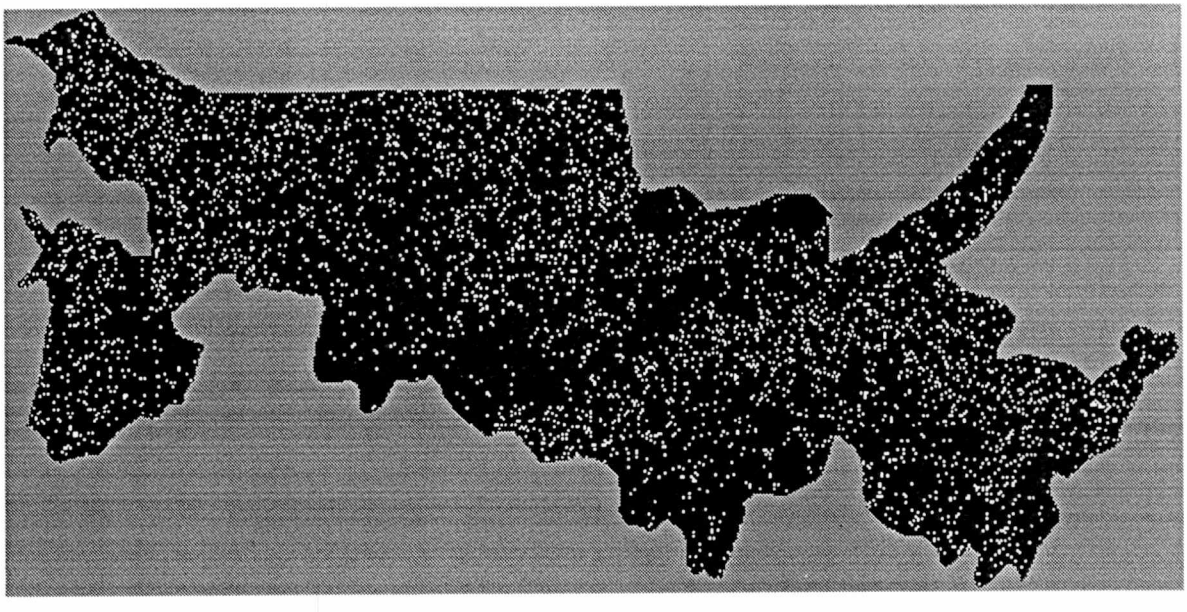

Figure 4.4: Parallel model ungulate location map for day 1.

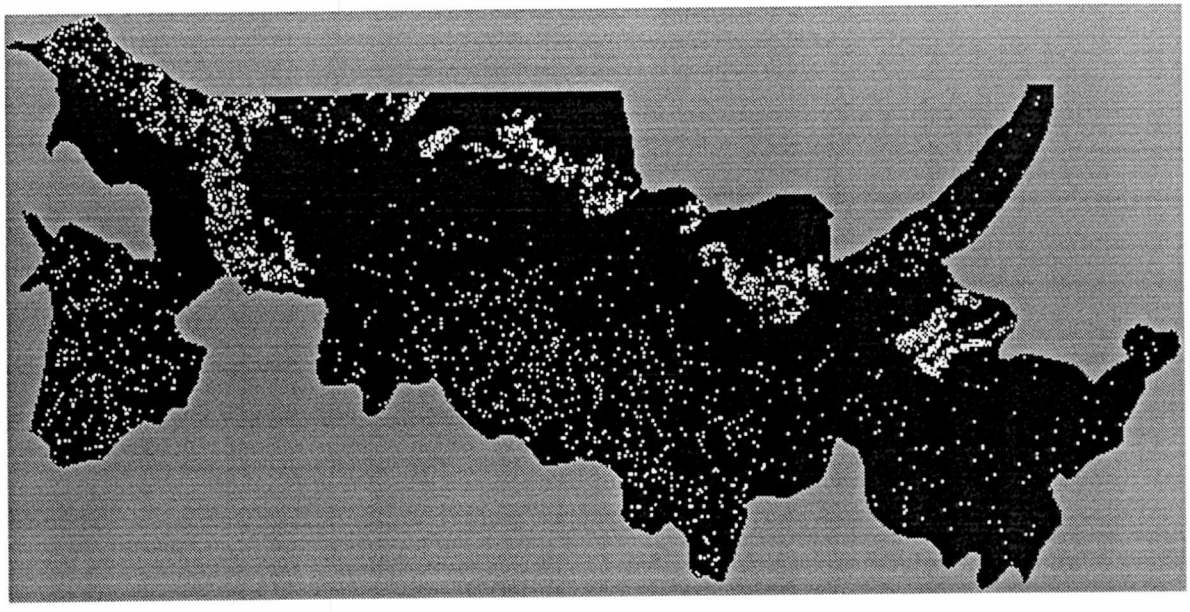

Figure 4.5: Sequential model ungulate location map for day 120.

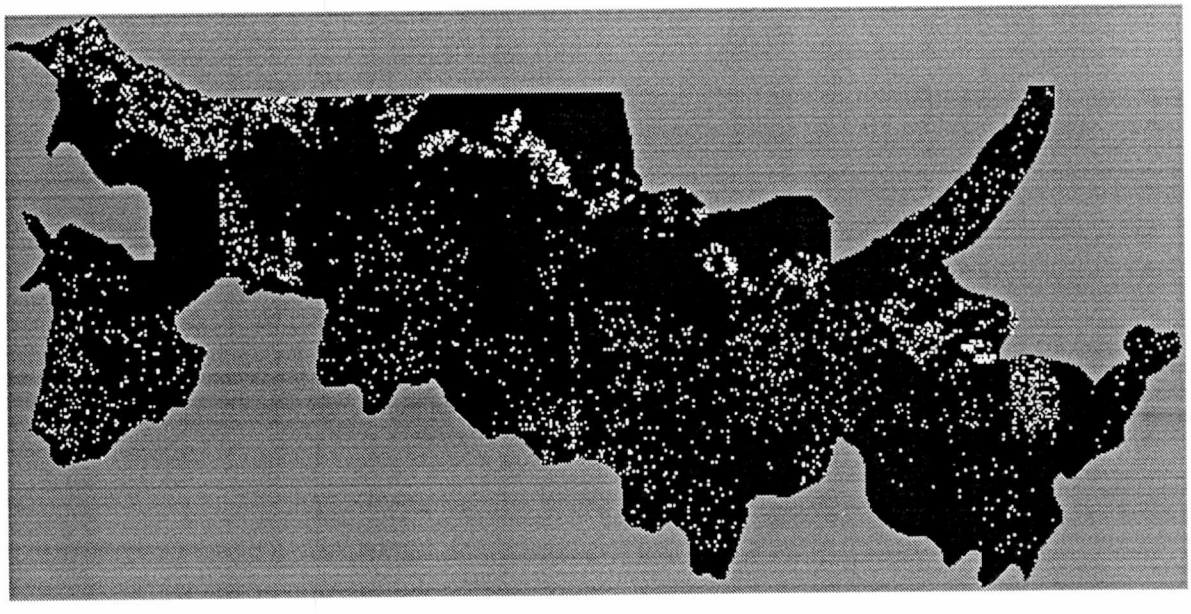

Figure 4.6: Parallel model ungulate location map for day 120.

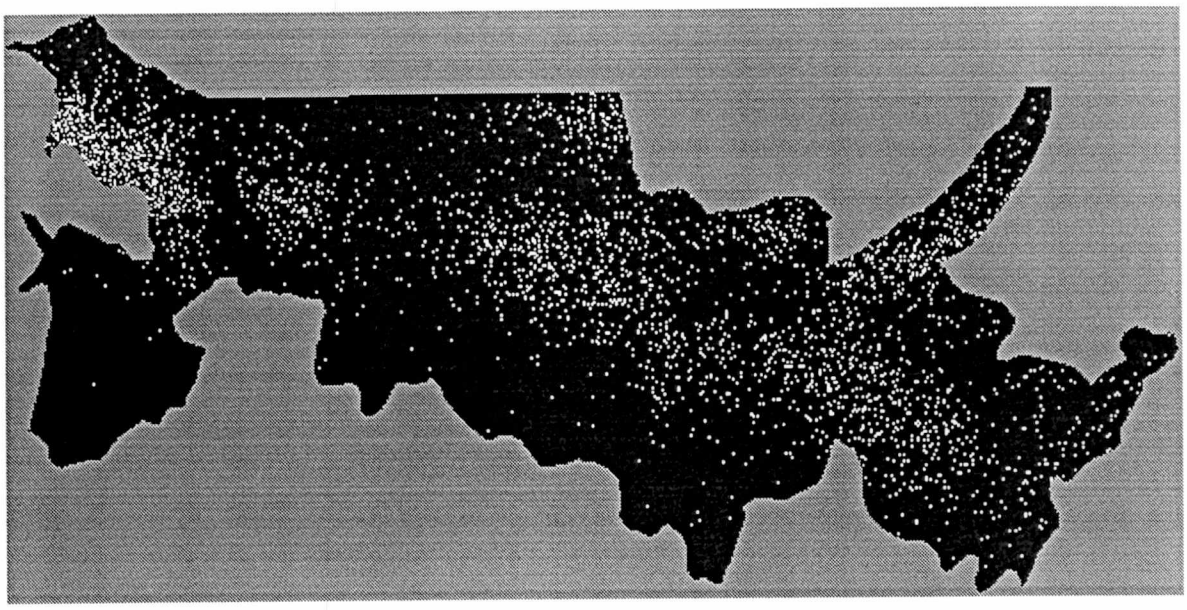

Figure 4.7: Sequential model ungulate location map for day 155.

Figure 4.8: Parallel model ungulate location map for day 155.

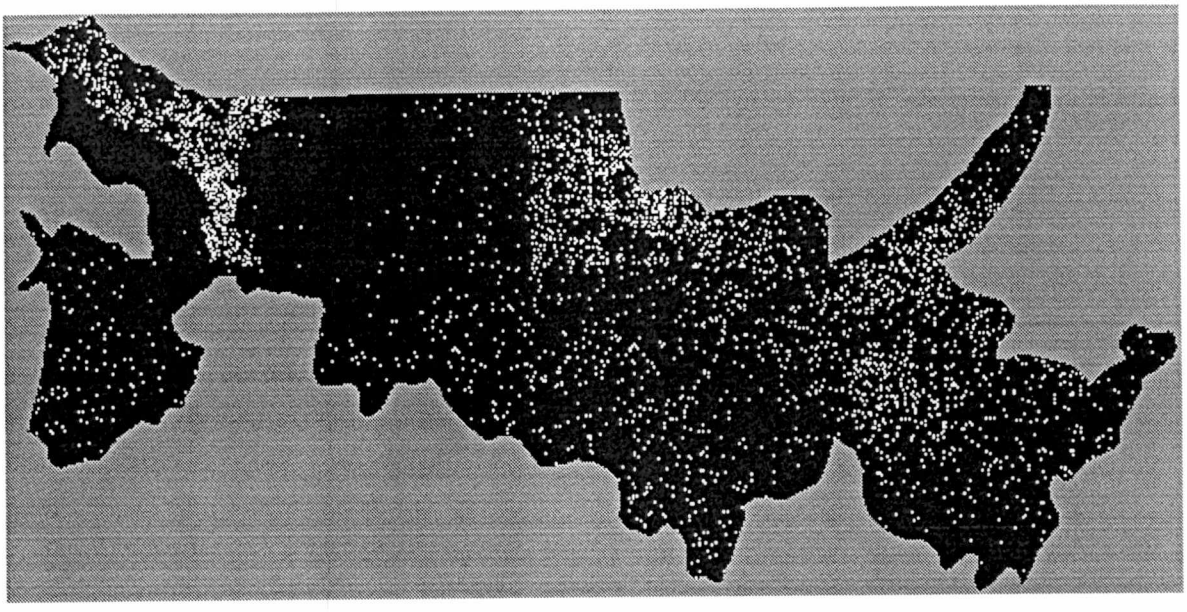

Figure 4.9: Sequential model ungulate location map for day 135.

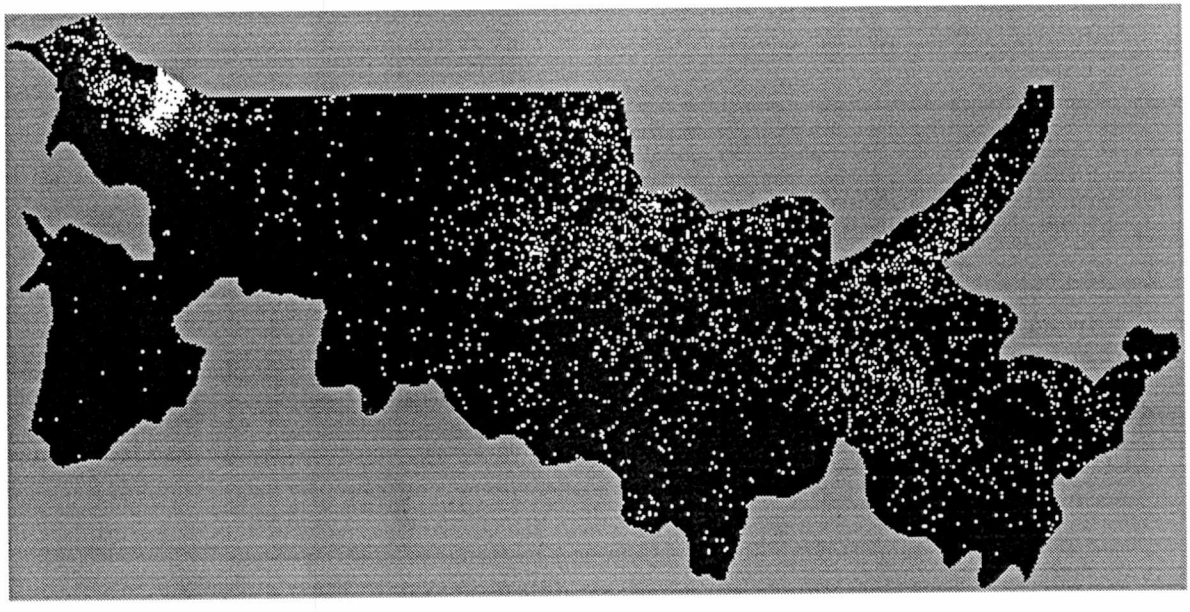

Figure 4.10: Sequential model ungulate location map for day 140.

Figure 4.11: Sequential model ungulate location map for day 145.

Figure 4.12: Sequential model ungulate location map for day 150.

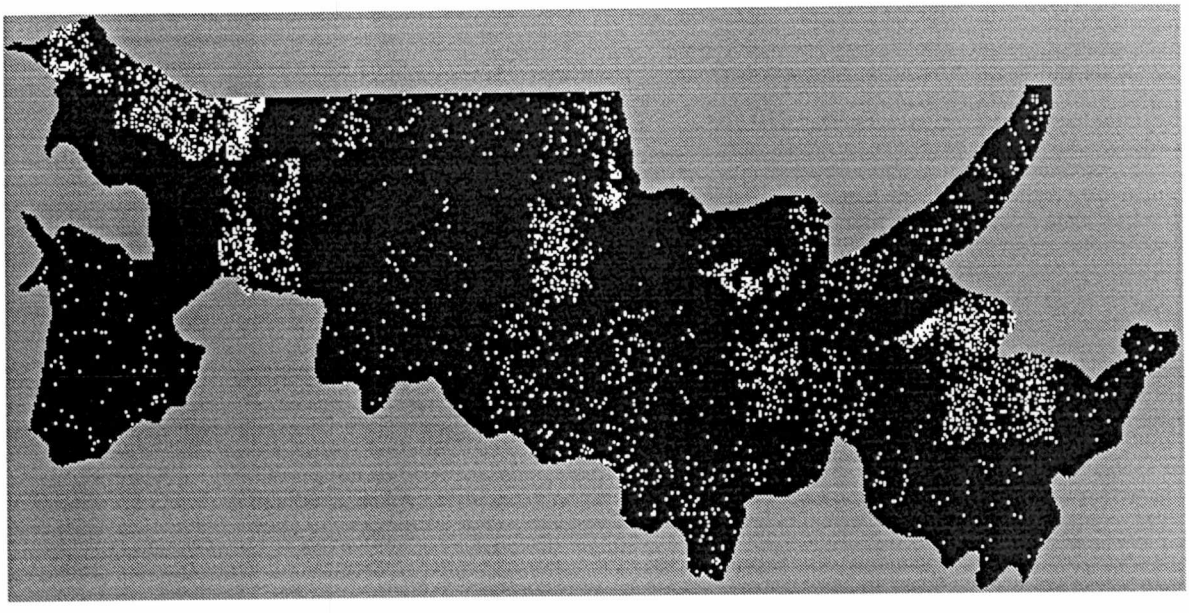

Figure 4.13: Parallel model ungulate location map for day 135.

Figure 4.14: Parallel model ungulate location map for day 140.

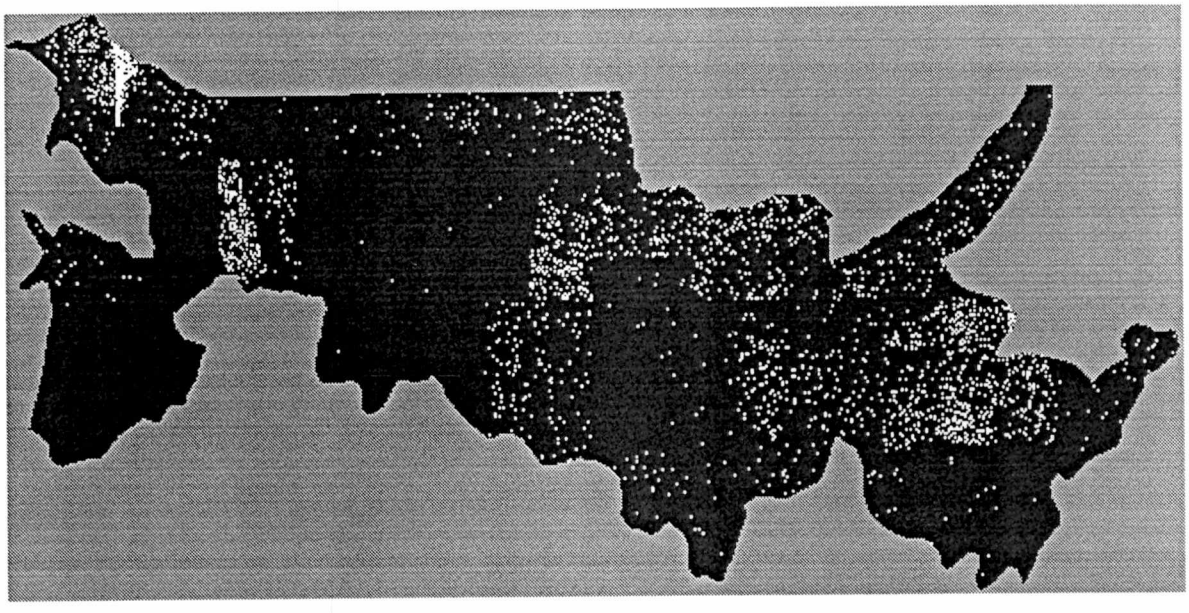

Figure 4.15: Parallel model ungulate location map for day 145.

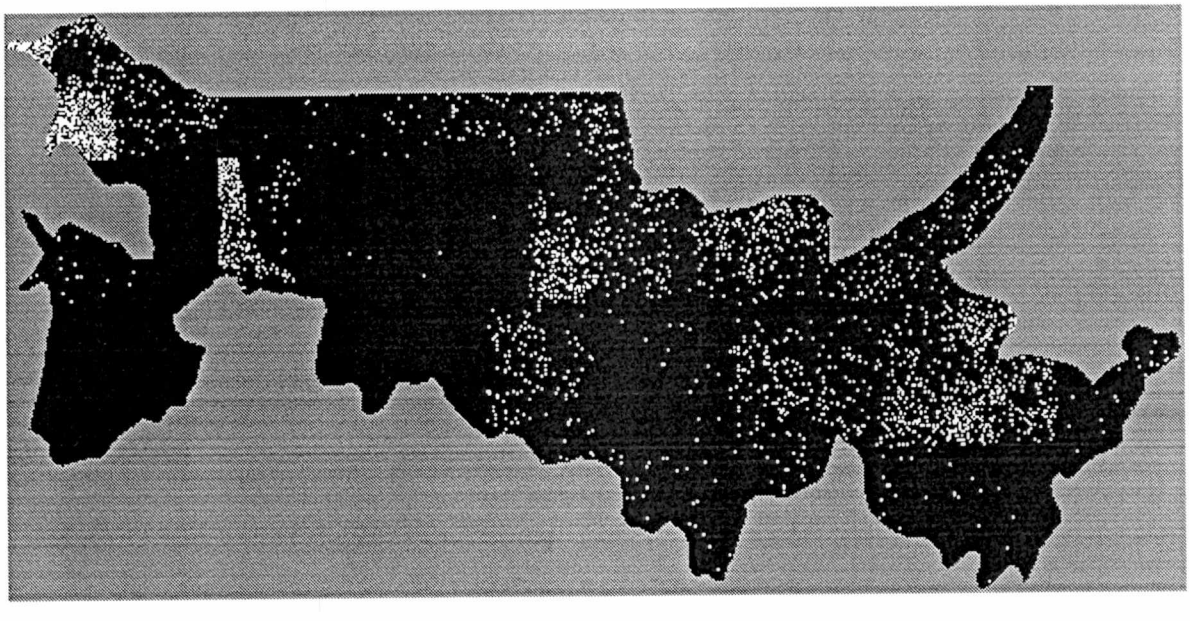

Figure 4.16: Parallel model ungulate location map for day 150.

4.1.2 Harmonic Mean

To obtain better agreement with the ungulate location maps from NOYELP, an alternative version of PNOYELP (HM), employing PMI harmonic means, was attempted. Harmonic means tend to be less sensitive than arithmetic means to large variances. On each landscape partition, the harmonic mean PMI values were generally smaller than the corresponding arithmetic PMI means calculated on each processor. The harmonic mean, m_h, for a landscape P is defined by

$$m_h = \frac{N}{\sum_{i,j \in P} \frac{1}{PMI_{i,j}}},$$

where N is the number of habitat pixels in P. Because the harmonic means calculated on each processor were smaller than the arithmetic means calculated in the AM version, a range of PMI harmonic mean threshold values (from 0.01 to 0.2) was used to determine suitability of remote processors for animal group movement. The PMI threshold value, as discussed earlier in this chapter, is the minimum PMI mean (PMI average) a neighboring processor must have in order for an animal group to move there. None of the values, however, improved the daily energy balance statistics and spatial resolution of the ungulate location maps. The animal groups were better distributed across the landscape in the ungulate location maps since they rarely left the processor they were assigned to at the beginning of the simulation. The daily energy balance statistics were also significantly worse in the HM version (the mortality rate almost doubled).

4.1.3 Arithmetic Mean with Percentage Threshold

A threshold version of PNOYELP (TH), which varies the PMI arithmetic mean threshold, was also implemented to determine the influence on the daily energy balance

statistics and ungulate location maps. This method allowed animal groups to move to another processor only if the receiving processor's PMI arithmetic mean was within a certain percentage of the sending processor's PMI arithmetic mean. For example, if the local processor's PMI arithmetic mean is 1.0 and the animal groups are only allowed to move to processors within 30% of the local processor's PMI arithmetic mean, the animal group attempting to move will only be allowed to move to a processor with a PMI arithmetic mean between 1.0 and 1.3. A range of percentages was used (0% to 100%) along with a range of PMI arithmetic mean thresholds (0.001 to 0.2) to determine suitability. However, this method showed no improvement over the original AM version. In one instance of the TH version, the PMI threshold was set to 0.1, thus the neighboring partition had to have a PMI arithmetic mean of at least 0.1 to be considered suitable. In addition, the neighboring partition's PMI arithmetic mean had to be within 30% of the local PMI arithmetic mean between day 130 and day 150. This combination returned daily energy balance statistics and ungulate location maps most similar to the sequential version (NOYELP). Because the groups could only move to processors with PMI averages within 30% of their own, they were prevented from moving directly to the partitions with the highest PMI average, and their progression was hence slower than the original sequential model.

4.1.4 Partial Averages

The final PNOYELP version (PA) computed a partial PMI average (arithmetic mean) over each half of the processor. Four averages were computed for each processor, one corresponding to each border of the partition (north, south, east, and west). The partial PMI averages were then broadcast to the 32 PNs as in the original parallel version. An animal group that decided to move to a new processor compared the

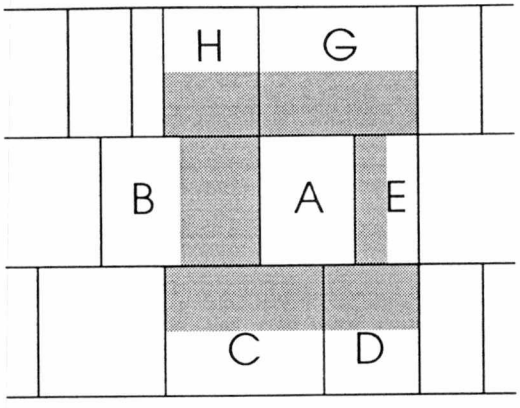

Figure 4.17: Partial averages compared by an animal group located on processor A.

averages corresponding to the shared border. As illustrated in Figure 4.17, an animal group located on processor A checks the shaded portions of its nearest neighbors (B, C, D, E, G, and H) only if they are within the maximum moving distance. The only noticeable difference between the PA and AM version was that more time was spent each day sending the three other averages to the remote processors. The daily energy balance statistics and ungulate location maps remained very similar.

4.2 Performance of Parallel Versions

Improving the performance of this model was a primary goal of this thesis. The sequential and parallel versions were compared based on speed improvement (i.e., how much faster the parallel version executed on the 32-processor CM5 relative to the sequential NOYELP program running on a SPARCstation 10). Both models were compiled using the current FORTRAN-77 SPARCcompiler (`f77` version 2.01) using the `-O` option for optimization. The parallel model was also linked with `cmmd-ld` version 3.2, the CMMD link editor, to create the CM5 executable files. Elapsed times recorded in seconds (wall-clock times) and speed improvements for PNOYELP

Table 4.1: Wall-clock times (in seconds) for the NOYELP and PNOYELP models with a varying number of elk. Speed improvements of each parallel version over the original sequential version are also listed.

Parallel version	Number of elk	Speed NOYELP	Speed PNOYELP	improvements
AM	6,000	1230	184	6.68
	18,000	2130	451	4.72
	30,000	3317	1168	2.84
HM	6,000	1230	188	6.54
	18,000	2130	281	7.58
	30,000	3317	494	6.71
TH	6,000	1230	184	6.68
	18,000	2130	388	5.49
	30,000	3317	1103	3.00
PA	6,000	1230	225	5.46
	18,000	2130	461	4.62
	30,000	3317	1146	2.89

are listed in Table 4.1. Each version of PNOYELP executed in substantially less time than the sequential model with a peak speed improvement of 7.6 times faster than the sequential version NOYELP running on a Sun SPARCStation 10. Considering a maximum possible speedup of 10.1 from Amdahl's law in Section 3.1, the speed improvements of the PNOYELP versions were quite reasonable. The speedup calculated by Amdahl's law required the animals not to move between processors, which is unrealistic and explains, at least in part, why the speed improvements of PNOYELP did not achieve the 10.1 maximum. The PNOYELP model incurred a substantial amount of communication and computation overhead moving animals between processors which is not accounted for in the sequential model. The difference in the number of animals used in a simulation greatly affected the speed improvements. As the number of animal groups increased, more animals were moving between processors, and the size of messages sent between processors grew proportionately. Also, the HM version was considerably faster than other versions since the animal groups tend to move between

processors less, while the PA version was somewhat slower due to the overhead of sending 3 additional PMI averages each day, as mentioned in Section 4.1.4.

4.3 Performance of PNs

A comparison of the computation time required on the PNs is an important determinate of how load-balanced the computation within the PNOYELP versions were. A balanced load ensures each PN will have about the same amount of computation while minimizing idle time. In this thesis, only actual computation time is recorded on each PN in an attempt to measure the load (Figure 4.18). The variance in time between PNs is, for the most part, due to the number of animal groups on each processor, but sending and receiving animal groups each day incurred additional overhead. Figure 4.19 illustrates the total number of animal groups a processor was responsible for over 5 simulations of the AM version. To create a more balanced computational load across the PNs, each PN would need to remain responsible for an equal number of animal groups (see Chapter 5).

In order to measure idle time on each node, an attempt to obtain a histogram of message-passing times was made. The elapsed time spent sending messages on each node was very similar, since for PNOYELP implementations synchronous communication was adopted. Unfortunately, CMMD timing routines do not offer a method of measuring idle times.

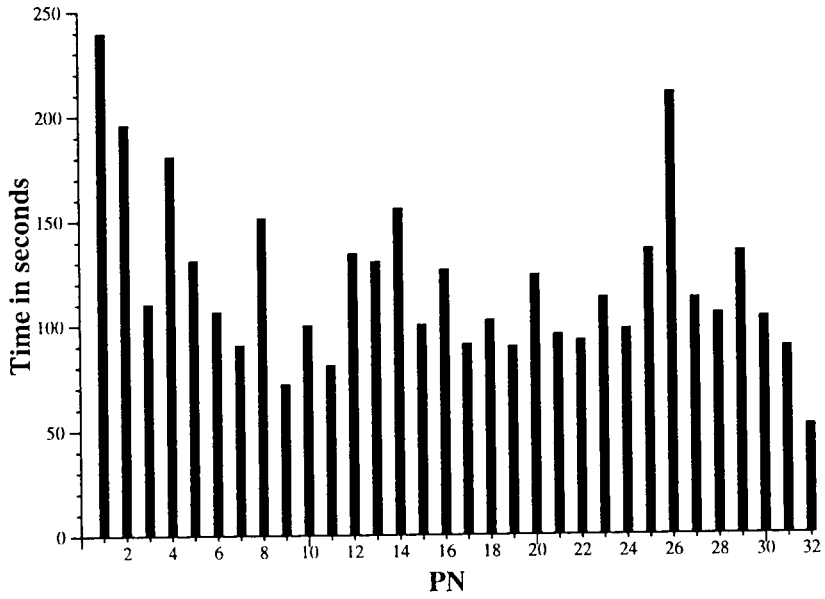

Figure 4.18: Total computation time on PNs for 5 repetitions of the AM version.

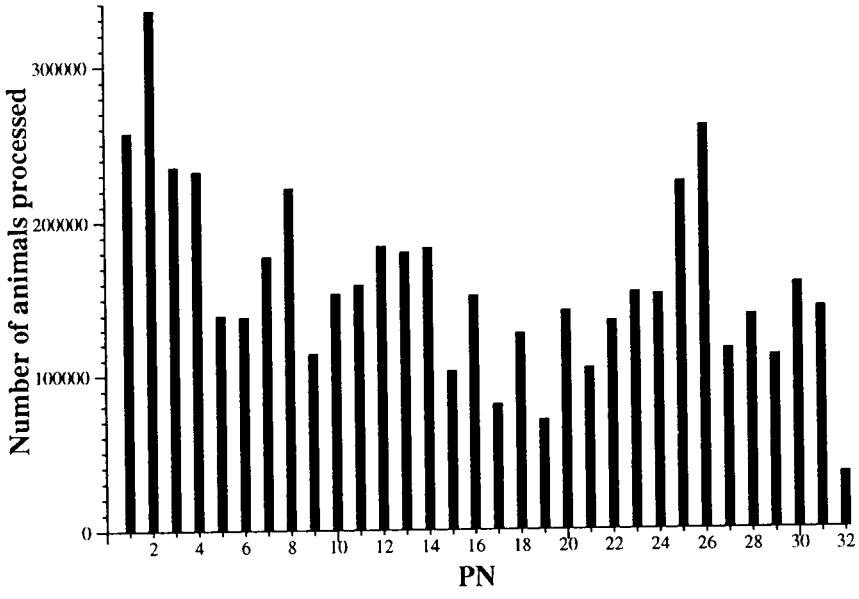

Figure 4.19: Total number of animals processed by each PN for 5 repetitions of the AM version.

4.4 Recommendations

Among the four PNOYELP versions, the AM version provided daily energy gain statistics closest to the sequential model NOYELP with a reasonable speed improvement. The ungulate location maps from the AM version represented the direction of animal movement very well, although the processor partition boundaries were visible in the maps. None of the PNOYELP versions were able to avoid this phenomenon. Although the other versions provided better speed improvement, their resulting survival rates dropped considerably lower than that of the sequential model because the animal groups were not traveling far enough to find suitable forage.

Chapter 5

Summary and Future Work

In this thesis, a parallel version of the sequential NOYELP model was implemented on the Thinking Machines CM5. Results were very promising with reasonable speed improvements and similar outputs between the original sequential model and the parallel model. Future work could provide better speed improvements and ungulate location maps that are more similar to the maps from the sequential model.

This chapter will discuss modifications to the PNOYELP model that could produce subsequent parallel models based on more of the assumptions of the original sequential model. These additions to the model, however, could impair the speed improvements realized in the current PNOYELP model.

As the animal groups in the model progress through the 180-day move/graze cycle, they tend to converge in specific regions of the landscape. At that time, the processors that own the landscape partitions where the animal groups converge have more work than other processors because a majority of time spent in the model is in the move/graze sequence of each animal group (as discussed in Chapter 2). If the partitions could be resized during the simulation to reflect the animal movement across

the landscape, a more balanced work load could be achieved on each processor. An assessment of the work needed to repartition the landscape would be needed to determine if time spent repartitioning in this model would be worthwhile. If the model consistently covered a larger landscape or if more animal groups were simulated, the repartitioning scheme would definitely be worthwhile; otherwise, if the landscape remained small with few animal groups, an optimal initial partitioning scheme, which was not altered during the simulation, would probably suffice.

In the design of a repartitioning scheme, a variable must be selected to trigger the repartitioning. In the NOYELP model, one of two variables could be chosen: the number of animals per processor, or the PMI average on a partition. The PMI average could be used to predict when and how many animal groups will move to a given area of the landscape. If a partition's average is substantially higher than surrounding partitions, more animal groups will move to that portion of the landscape once they do not find suitable pixels on their local partition. The influx of groups will then increase the work load because the partition will be responsible for a disproportionate number of groups. Once a partition with a high PMI average is recognized, the daily move/graze sequence would be halted and local copies of the ungulate location map, available biomass map, total biomass map, and snow condition map could be sent to the host, and new maps with current conditions over the 32 PNs would be created. Ungulate specific arrays (current body weight, row and column location of animal groups, and calf groups associated with cow groups) will also be recreated on the host. The host program will then redefine the partition boundaries for each PN by moving the column boundaries until a more reasonable PMI average is attained for each partition. Once new boundaries are determined, the host will broadcast the new maps and ungulate specific arrays. Finally, the PNs will determine new

nearest neighbors and which ungulate groups they are responsible for and the daily move/graze sequence will continue. This method of repartitioning the landscape (*host partitioning*) depends on the host program to evaluate the landscape and determine a new partitioning scheme. Since the repartitioning of the landscape could occur one or many times during the simulation to achieve a more balanced work load, an evaluation of the cost of repartitioning would be valuable in determining the number of times to repartition.

Using the PMI average to determine when to repartition the landscape is a method that attempts to prevent an imbalanced work load before animal groups have converged on a portion of the landscape. With this method, there is no guarantee the load will be balanced across the processors. Instead, if the number of animal groups on each partition is used to determine when to repartition the landscape, a balanced work load could be guaranteed for the day the landscape is repartitioned. The benefit of using the number of animal groups per processor to determine when and how to repartition the landscape is that the programmer can determine which processor has a disproportionate amount of work and repartition accordingly. If the PMI average is used to determine when and how to repartition, the programmer can only attempt to predict where the animals will congregate, but it is not a guarantee the ungulate groups will move to that processor.

In the *host partitioning* scheme discussed previously, the host PN is responsible for repartitioning the landscape and updating every PN. Another option could require the PNs to exchange information between themselves (*node partitioning*) and not involve the host PN. Once a partition's PMI average or the number of animal groups is above a given threshold, the PN would exchange messages with nearest neighbors in order to identify a partition with fewer than the average number of animal groups or

a significantly lower PMI average. The neighbor chosen with a smaller work load will accept portions of the landscape from the overloaded PN in an attempt to decrease the work load. The information exchanged between processors is determined by the variable chosen (number of animal groups or PMI average) to determine when to repartition. If the number of animal groups is the threshold variable, the two neighboring processors would need to exchange the ungulate location map; otherwise, if the PMI average is the threshold variable, the neighboring processors would need to exchange either the PMI values at each pixel, or the three data layers necessary to calculate the PMI average. Once the information is exchanged, the two processors would adjust their partition boundaries until either the number of animal groups or the PMI average on each partition was more balanced. The new boundaries would be broadcast to all other PNs and the nearest-neighbors partitions would be recalculated. The PN receiving a new portion of the landscape would also need all dynamic ungulate-specific information about the groups in the landscape portion along with the dynamic data layers now included within its partition boundaries.

The *node partitioning* scheme described previously would considerably reduce the amount of communication because only nearest neighbors would be exchanging information instead of every PN communicating with the host. This method may also be able to better exploit the fat tree architecture of the CM5.

Bibliography

Bibliography

[BCM94] M. Berry, J. Comiskey, and K. Minser. Parallel Analysis of Clusters in Landscape Ecology. *IEEE Computational Science and Engineering*, 1(6):24–38, Summer 1994.

[Com93] E.J. Comisky. Data-Parallel Implementations of Map Analysis and Animal Movement for Landscape Ecology Models. Technical Report CS–93–207, University of Tennessee, Knoxville, Tennessee, August 1993.

[Cor93] Thinking Machines Corporation. *Connection Machine CM-5 Technical Summary.* Thinking Machines Corporation, Cambridge, Massachusettes, 1993.

[Dia79] H.F. Diaz. Ninety-one Years of Weather Records at Yellowstone National Park, Wyoming, 1887-1997. Technical report, National Oceanic and Atmospheric Administration, Environmental Data and Information Service, National Climatic Center, Asheville, North Carolina, 1979.

[DM82] R.A. Dirks and B.E. Martner. The Climate of Yellowstone and Grand Teton National Parks. Occasional paper number 6, U.S. National Park Service, Washington, District of Columbia, 1982.

[DRSS89] D. Despain, A. Rodman, P. Schullery, and H. Shovic. Burned Area Survey of Yellowstone National Park: The Fires of 1988. Technical report, Division of Research and Geographic Information Systems Laboratory, Yellowstone National Park, Wyoming, 1989.

[GOT92] R. Gardner, R.V. O'Neill, and M.G. Turner. *Humans as Components of Ecosystems: Subtle Human Effects and the Ecology of Populated Areas.* S.T.A. Pickett and M.J. McDonnell, Springer-Verlag, New York, 1992.

[HGT+93] W.W. Hargrove, R.H. Gardner, M.G. Turner, W.W. Romme, and D.G. Despain. Simulating Fire Patterns in Heterogeneous Landscapes: the analysis and interpretation of landscape heterogeneity. preprint, 1993.

[Hwa93] Kai Hwang. *Advanced Computer Architecture: Parallelism, Scalability, Programmability.* McGraw-Hill, Inc., New York, New York, 1993.

[TWW+93] M.G. Turner, Y. Wu, L.L. Wallace, W.H. Romme, and A. Brenkaert. Simulating Interactions Among Ungulates, Vegetation and Fire in Northern Yellowstone National Park During Winter. *Ecological Applications*, 1993. Submitted.

Appendices

Appendix A

NOYELP Specifics

A.1 Outline of NOYELP Subroutines

The serial NOYELP model is implemented as a 4736 line Fortran-77 program, divided into 13 subroutines.

`subroutine input()`

lines: 304

Subroutine `input()` inputs the parameter values, snow index matrix, habitat type matrix, and interface choices, and sets initial rows and columns for bison distribution.

parameters: none

called by: `main()`

calls to to: none

`subroutine initial()`

lines: 701

Subroutine `initial()` initializes biomass and ungulate location matrices and establishes the chosen pattern of burned areas on the landscape.

parameters: none

called by: `main()`

calls to to: `unevenb()`, `ranmap()`

`subroutine unevenb()`

lines: 40

Subroutine `unevenb()` creates an uneven biomass map for the landscape.

parameters: none

called by: `initial()`

calls to: none

`subroutine ranmap()`

lines: 63

Subroutine `ranmap()` creates a random habitat map for the landscape, if random landscape is chosen.

parameters: none

called by: `initial()`

calls to: none

`subroutine ssnow()`

lines: 174

Subroutine `ssnow()` distributes snow depth in each pixel and updates it at the start of each snow period (3 days).

parameters: none

called by: `main()`

calls to: none

`subroutine ungfind()`

lines: 153

Subroutine `ungfind()` locates each ungulate group on the landscape map, then initiates moving and grazing for the day.

parameters: none

called by: `main()`

calls to: `graze()`, `move()`

`subroutine graze(i,j,ungrank)`

lines: 128

Subroutine `graze()` allows an ungulate in a resource pixel to forage. Total biomass on the grazed pixel is reduced by the amount consumed by the grazing ungulate group. Foraging is precluded on a given site when biomass declines below a *refuge* level.

parameters:		
	integer `i`	row (x-coord.) of current group
	integer `j`	column (y-coord.) of current group
	integer `ungrank`	rank of current group in movement series

called by: `ungfind()`

calls to: `move()`, `energet()`

`subroutine move(i,j,ungrank)`

lines: 277

Subroutine `move()` implements the destination resource pixel selection phase of the movement rule.

parameters:	integer `i`	row (x-coord.) of current group
	integer `j`	column (y-coord.) of current group
	integer `ungrank`	rank of current group in movement series

called by: `graze()`, `ungfind()`

calls to: none

`subroutine energet(i,j,ungrank)`

lines: 169

Subroutine `energet()` calculates energy cost and gain each day for each ungulate. Body weight lost is tracked and mortality is simulated.

parameters:	integer `i`	row (x-coord.) of current group
	integer `j`	column (y-coord.) of current group
	integer `ungrank`	rank of current group in movement series

called by: `graze()`

calls to: none

`subroutine patch()`

lines: 350

Subroutine `patch()` identifies clusters of available biomass pixels and reports the number of clusters, average cluster size, and largest patch size for available resources.

parameters: none

called by: `main()`

calls to: none

`subroutine statist()`

lines: 1018

Subroutine `statist()` provides a statistical summary over all replicate runs and calculates mean, minimum, maximum values and standard deviation for each replicate at day 180.

parameters: none

called by: `main()`

calls to: none

`subroutine outfile()`

lines: 109

Subroutine `outfile()` writes summary results to an outfile.

parameters: none

called by: `main()`

calls to: none

A.2 Selected Formulas used in NOYELP model computations

Feedback due to foraging: represents the effect of a reduction in the amount of available forage on the rate of forage intake.

$$FBforage = 1 - \frac{REF}{BIO}$$

where:

FB_{forage} = value ranging from 0–1 (unitless),

REF = refuge value of biomass not available to ungulates (kg/ha), and

BIO = actual biomass in that pixel (kg/ha).

Feedback due to snow: represents the effect of snow on the ability of an animal to forage.

$$FB_{snow} = \left[1 - \frac{(SWE - SWLO)_+}{SWHI - SWLO}\right]_+$$

where:

FB_{snow} = value ranging from 0–1 (unitless),

SWE = actual snow water equivalent in that grid cell,

$SWLO$ = the SWE value at which limitation to foraging begins,

$SWHI$ = the SWE value at which feeding goes to zero, and

+ indicates that the term must remain positive, i.e., it is set to zero if negative.

DIST: maximum daily moving distance as modified by snow conditions; used to constrain animal movement such that elk or bison move a shorter distance when

snow conditions are severe.

$$DIST = \frac{MDISTM}{1 + Y/100}$$

where:

$DIST$ = the modified maximum daily moving distance in snow,

$MDISTM$ = the initial maximum daily moving distance (category-specific) and,

Y = the relative increase in travel energetics in snow.

Energy balance equations:

$$E_{balance} = E_{gain} - E_{cost}$$

$$E_{gain} = I * ENPK$$

$$E_{cost} = E_{maint} - E_{travel}$$

where:

I = total intake of forage(kg), $ENPK$ = forage energy content (kcal/kg),

E_{maint} = metabolizable energy needed for zero energy balance, excluding travel (kcal), and

E_{travel} = energy cost of travel (kcal).

Energy content of forage:

$$ENPK = GE * IVDMD * MC$$

where:

$ENPK$ = forage energy content (kcal/kg),

GE = gross energy (4400 kcal/kg),

$IVDMD$ = in vitro dry matter digestibility (0.374 from field data), and

MC = metabolizable energy coefficient (0.82).

E_{maint}: Maintenance energy

$$E_{maint} = ME * BW^{0.75}$$

where:

ME = metabolizable energy needed per kg body weight (kcal/kg), and

BW = present body weight (kg) of the ungulate.

E_{travel}(no snow): Energy cost of traveling in the absence of snow.

$$E_{travel}(nosnow) = [2.97kcal/kg * BW^{-0.34}] * BW * S$$

where:

S = distance traveled (km), and

BW = present body weight of the animal (kg).

Y: relative increase in energy costs for travel in snow (%)

$$Y = [0.71 + 2.6(\rho - 0.2)] * RSD * e^{0.019+0.016(\rho-0.2)] * RSD}$$

where:

ρ = snow density (g/cm^3), and RSD = relative sinking depth

[(sinking depth/brisket height) $\times$ 100].

Appendix B

PNOYELP Subroutines

B.1 Outline of PNOYELP Subroutines

The parallel PNOYELP model is implemented as a two programs: a 3039 line FORTRAN-77 host program and a 2014 line FORTRAN-77 node program. This appendix only discusses the node program and node program subroutines that differ from the original sequential NOYELP model.

`main-node()`

lines: 1056

`main-node()` is the driver for the program running on the PNs and performs all daily communication with the host program and with the other PNs for animal movement between partitions. After initializing each PN program, it locates each ungulate group on the landscape map, then initiates moving and grazing for the day. Once the move/graze cycle for all present ungulate groups is complete, `main-node()` will accept new ungulate groups from nearest-neighbor PNs and complete their move/graze cycle. It will also send ungulate groups to other neighboring partitions so that they can complete their daily cycle on another partition.

parameters: none

called by: `none()`

calls to: `graze()`, `move()`

`subroutine ssnow()`

lines: 154

Subroutine `ssnow()` runs on each of the 32 PNs. It distributes snow depth in each pixel of the PN's landscape partition and updates it at the start of each snow period (3 days).

parameters: none

called by: `main-node()`

calls to: none

`subroutine graze(i,j,ungrank)`

lines: 159

Subroutine `graze()` allows an ungulate in a resource pixel on the PN's landscape partition to forage. Total biomass on the grazed pixel is reduced by the amount consumed by the grazing ungulate group. Foraging is precluded on a given site when biomass declines below a *refuge* level.

parameters:		
	integer `i`	row (x-coord.) of current group
	integer `j`	column (y-coord.) of current group
	integer `ungrank`	rank of current group in movement series

called by: `main-node()`

calls to: `move()`, `energet()`

`subroutine move(i,j,ungrank)`

lines: 415

Subroutine `move()` implements the destination resource pixel selection phase of the movement rule. The ungulate group will select a resource pixel on the PN's partition, select a new nearest-neighbor partition to occupy, or be placed randomly within it's maximum moving distance on the PN's partition. If a new partition is selected, the daily move/graze cycle for the ungulate group will be halted and specific ungulate information saved so that the ungulate group will be able to continue the cycle on the new partition at the end of the day.

parameters:	integer `i`	row (x-coord.) of current group
	integer `j`	column (y-coord.) of current group
	integer `ungrank`	rank of current group in movement series

called by: `graze()`, `main-node()`

calls to: none

`subroutine energet(i,j,ungrank)`

lines: 150

Subroutine `energet()` calculates energy cost and gain each day for each ungulate present on the PN's partitions. Body weight lost is tracked and mortality is simulated.

parameters:	integer `i`	row (x-coord.) of current group
	integer `j`	column (y-coord.) of current group
	integer `ungrank`	rank of current group in movement series

called by: `graze()`

calls to: none

Appendix C

Daily Energy Balance Statistics

Daily Energy Balance Statistics
Total Biomass in the Landscape (kg)
Available Biomass in the Landscape (kg)
Daily Biomass Consumed (kg)
Daily Body Weight Lost (per ungulate group)
Daily Travel Energy Cost (per ungulate group)
Daily Travel Distance (meters/ungulate group)
Number of Resource Patches (high and low resource)
Average Patch Size (high and low resource)
Largest Patch Size (high and low resource)
Total Biomass in Habitat Type (kg/ha)
Available Biomass in Habitat Type (kg/ha)
Survival (per ungulate group)

Note: The boldface statistics were chosen to compare the sequential (NOYELP) and parallel (PNOYELP) models.

VITA

Ember Uziel was born November 13, 1967 in Oak Ridge, Tennessee. She was graduated from Oak Ridge High School in 1986 and received the Bachelor of Science degree in Math/Computer Science in 1991 from Oglethorpe University in Atlanta, Georgia. She is married to Jeff Constantin, whom she met in graduate school, and was awarded the Master of Science degree in Computer Science in December 1994.