A Parallel Fish Landscape Model for Ecosystem Modeling

Dali Wang [1], Michael W. Berry
Department of Computer Science
203 Claxton Complex
University of Tennessee, Knoxville, TN 37996, USA.
Email: [dwang,berry]@cs.utk.edu

Eric A. Carr, Louis J. Gross
The Institute for Environmental Modeling,
569 Dabney Hal
University of Tennessee, Knoxville, TN 37996, USA.
Email: [carr, gross]@tiem.utk.edu

Keywords: Parallel Simulation, Computational Ecology, Dynamic load balancing, Ecosystem Modeling

Abstract

Parallelization of a landscape fish population model (ALFISH) is an important effort towards high performance Across Tropic Level System Simulation (ATLSS) on a computing grid. ALFISH models the impacts of different water management strategies in the South Florida region on the fresh water fish population, providing estimates of the food resource available to wading birds. The parallel ALFISH model delivers similar results as those from a sequential implementation. Compared with the average simulation time of the sequential model, which is about 35 hours, the speed improvement of the parallel model on a symmetric multi processor (SMP) is substantial. Using 14 processors, the runtime of the parallel model with static partitioning is less than 4 hours, and that of the parallel model with dynamic load-balancing is less than 3 hours.

1 Introduction

The landscape of South Florida is a complex environment that has been subjected for decades to extensive modifications through alterations of the natural hydrologic flows in the region. Disruptions in the natural flows have been the catalyst for profound changes in the vegetation and animal life in the region. Attempts are now being made to repair the devastating effects of these changes in water flow on the ecosystem of the South Florida region [1]. The effects of these corrections are modeled to analyze alternative detailed management plans and assist in the planning process. The most effective way to evaluate the effects of this complex environment is through computer modeling [2, 3]. The Across Trophic Level System Simulation (ATLSS), a family of linked models, was developed to address this regional environmental problem including components spanning a wide variety of spatial, temporal and organismal scales.

[1] corresponding author

1.1 ATLSS Model Overview

ATLSS is a multiscale ecological multimodel designed to assess the effects on key biota of alternative water management plans for the regulation of water flow across the Everglades landscape. Models included are spatially-explicit, accounting for heterogeneity across the landscape. ATLSS is illustrated schematically in Fig. 1.1. ALFISH is an Intermediate Trophic Level Functional Groups

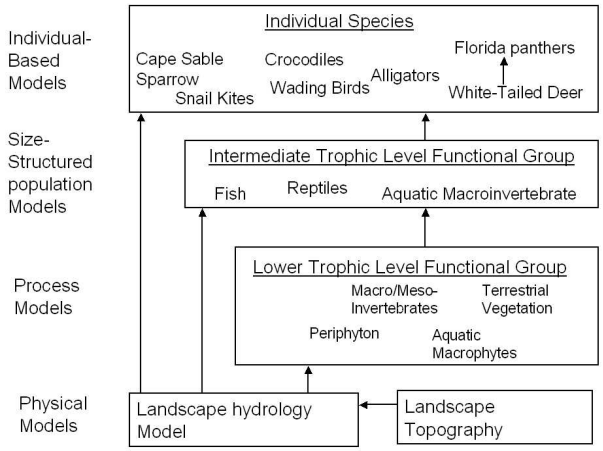

Figure 1: Schematic of the Across Trophic Level System Simulation (ATLSS) approach for modeling the ecosystem of South Florida.

model which includes two main subgroups (small planktivorous fish and large piscivirous fish), structured by size. Arrows represent the direction of effects from abiotic forces and lower and intermediate trophic levels to the higher trophic levels. Feedback effects of higher trophic levels on lower and intermediate levels are also considered. This paper is particularly focused on the enhancement of one of these models, i.e, the fish component. The fish model is an important contribution to the higher-level landscape model, since it provides a food base for several wading bird models [4]. An objective of the ALFISH model is to compare, in a spatially explicit manner, the relative effects of alternative hydrologic scenarios on fresh-water fish densities across South Florida. Another objective is to provide a measure of dynamic, spatially-explicit food resources available to wading birds.

The ALFISH model has been developed in part to integrate with two wading bird models, a proxy individual-based wading bird (WB) model [5] and a Spatially-Explicit Species Index (SESI) wading bird model [4]. Due to the multi-scale natural of the ATLSS models, this integration proposed challenges to modelers. For example, in the WB model, the timestep is set as 15 minutes in order to capture the basic behaviors of wading birds. In the ALFISH model, the timestep is 5 days. In order to compare the relative potential for breeding

and/or foraging across the landscape without tracking in detail the population dynamics or behavior of individual wading bird, the results of SESI wading bird are represented yearly. A major concern associated with integrating ALFISH into ATLSS architecture using computing grid has been its runtime. The average runtime is 35 hours on a 400MHz (Ultra Sparc II-based) Sun Enterprise 4500 for a typical 31-year simulation. One objective of this parallelization is to speedup the ALFISH execution for its practical use within ATLSS by allowing real-time data generation and information sharing between multiple ATLSS models. With practical speedup we can also begin to make available a broadening parameter field for scenario runs and increase remote accessibility for natural resource managers at various South Florida agencies, through web interfaced grid computing [6] in a manageable time frame.

1.2 Notation

In the following sections, m denotes meters and $C_{i,j}$ is used to refer to a grid cell containing information about an area that is 500m by 500 m. The subscripts i and j indicate the location of the grid cell on a landscape map. Similarly C_i is used to refer to a row of grid cells in the landscape map. For the two-layer communication model used in the parallel model, P_i is used to represent processor ID number, and P_0 always refers to the master processor which is used to collect data from other processors and to control the output operations.

1.3 Computational Environment

The computational platform used in this research was a Sun Enterprise 4500, configured with 14 400MHz Sun Ultra Sparc II processors, 10 GB memory and 3GB/s interconnections. This symmetric multiprocessors (SMP) is one of the main grid clusters comprising the Scalable Intercampus Research Grid (or SInRG)[7] at the University of Tennessee, Knoxville. An implementation of the MPI standard library[8], LAM-MPI, was selected to support the message-passing communication in the parallel ALFISH model.

2 ALFISH Model

2.1 Model Structure

The study area for ALFISH modeling contains 26 regions as determined by the South Florida Water Management Model[9, 10]. A complete list of these regions is provided in Fig. 2. The total area of Everglades modeled in ALFISH contains approximately 111,000 landscape cells, with each cell 500m on a side. Each landscape cell has two basic types of area: marsh and pond. The difference between marsh and pond areas is that the latter always is considered wet (contains standing water) regardless of any available water data. In the marsh area of each cell, there is a distribution of elevations based upon a hypsograph [4]. This hypsograph is used to determine the fraction of the marsh area that is still

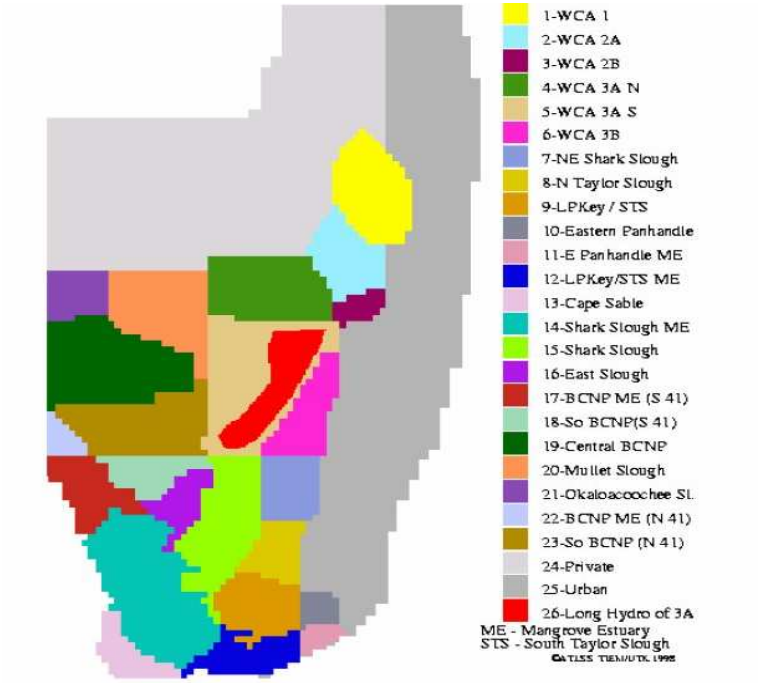

Figure 2: Subregions used by the ALFISH model

under water at a given water depth. A *pond* refers to permanently wet areas of small size, such as alligator holes, which are a maximum of 50 m^2 or 0.02% of the cell. The fish population simulated by ALFISH is size-structured and is divided into two functional groups: small and large fishes. Both of these groups appear in each of the marsh and pond areas. Each functional group in each area is further divided into several fish categories according to age, referred to as $ageClass$, and each age class has 6 size classes, referred to as $sizeClass$. The fish population in each cell is summarized by the total fish density (biomass) within that cell. Each cell, as an element of the landscape matrices, contains an array of floating-point numbers representing individual fish density of various age classes. The length of the array corresponds to the number of age classes for that functional group. Normally, when a fish density is referenced, the value reflects the total fish densities of all the fish age classes combined. Fig. 3 shows the simple age and size categorization of the fish functional groups in four landscape matrices.

In ALFISH, spatial and temporal fluctuations in fish populations are driven by a number of factors, especially the water level. Fluctuations in water depth, which affect all aspects of the trophic structure in the Everglades area, are provided through an input hydrology data file for each timestep throughout the execution of the model. Another landscape matrix, *region area matrix*, is used

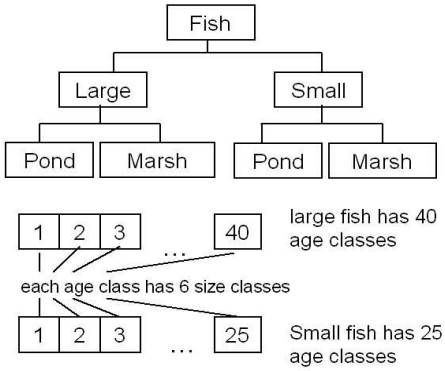

Figure 3: Simple fish functional group categorization

to determine which areas are included in the current simulation. The element of this region area matrix has a boolean value, specifying whether the element is included or not.

2.2 Fish Dynamics

The basic behaviors of fish simulated in the model include *density-independent fish movement, diffusive fish movement, mortality, growth and reproduction.* The general data flow of ALFISH is shown in Fig. 4.

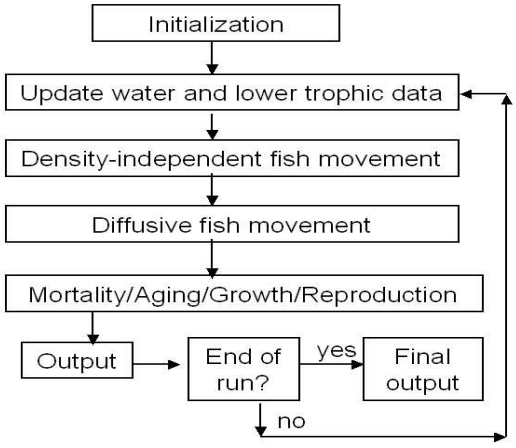

Figure 4: Flowchart diagram for the ALFISH model

2.2.1 Density-Independent Fish Movement

This function is designed to simulate the movement of fish between the marsh and the pond areas (if the cell has a pond area), and to neighboring cells, caused by drying out and reflooding. First, the water depth required by each size class, $depthSize(s)$, in order for that size class to survive is calculated by (see Gaff [4])

$$length(s) = \alpha_1[1.0 - exp(5\alpha_2[s+1] - \alpha_3)], \tag{1}$$

$$depthSize(s) = \frac{\beta_1 \times length(s)}{\beta_2 + length(s)}, \tag{2}$$

$$s = ageClass \times 6 + sizeClass - 1, \tag{3}$$

where s is the size class, α_1, α_2, β_1, and β_2 are constants, empirically deprived from field data. Then the model calculates the largest size, referred to as $maxSize$, of fish that can survive in the new water depth and compares this value to the old value from the previous timestep. The value of $maxSize$ depends on both the age classes and size classes. Specifically,

$$maxSize = \max_{\substack{water(i,j) \geq \\ depthSize(s)}} (s), \tag{4}$$

where $water(i,j)$ is the water depth of cell $C_{i,j}$, s is size class, and $depthSize(s)$ is the water depth derived from Eq. (2). At the current timestep, if the water depth of the cell has increased, and the cell has pond area, an appropriate fraction of fish (of sizes previously too large) is moved from the pond areas into the marsh area. If the cell water has decreased, some fish will not survive in that cell. In this case, an appropriate fraction of those fish are moved from marsh area into the pond area, another fraction of those fish are moved to adjacent cells, and the remaining portion of those fish are eliminated.

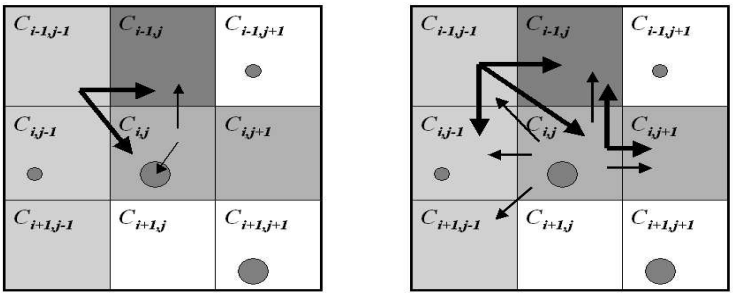

Figure 5: Calculation of fish movements from the center cell and the top-left cell. Left: density-independent fish movement. Right: diffusive fish movement. Circular regions refer to pond area. The shading of cells represents different water depths and arrows represent the movements of fish

The left plot of Fig. 5 illustrates the density-independent fish movement of three different fish classes(small/medium/large), that is $s = 3$, instead of the several dozen size classes defined by Eq. (3) in ALFISH. The shade of each cell represents water depth. White indicates no water in the marsh areas of the cell. Light/medium/dark gray indicates that the water depth in the cell is suitable for small/medium/large size fish to survive. Small circular regions represent pond areas. For the purpose of demonstration, let us assume that, small, medium and large size fish exist in the central cell $C_{i,j}$. At the present timestep, water depth decreases to the medium gray level, so a fraction of the large size fish in this cell should move to pond area within the cell. Another fraction of the large fish move to the adjacent heavy gray cell, $C_{i-1,j}$ where water depth is suitable for the large size fish to survive. Thin arrows in the left plot of Fig. 5 represent these movements. All remaining large fish in the central cell die at this timestep. Since the water depth in the central cell is still suitable for the small and medium size fish, those fish will remain. Note that this kind of movement is fish-density independent. Similarly, we assume that small and medium size fish exist in top-left cell, $C_{i-1,j-1}$. At the present timestep, the water depth drops so that only the small size fish can survive, the medium size fish must move to either cell $C_{i-1,j}$ or $C_{i,j}$. Bold arrows, in the left plot of Fig. 5, represent these movements.

2.2.2 Diffusive Fish Movement

This function is designed to simulate the movement of fish between adjacent cells mainly due to the relative differences in the water depth and fish densities. Movement into or out of a cell is assumed to occur only when the cell is more than 50% flooded. The following equations are used to determine the amount of fish to be moved out of the cell $c_{i,j}$:

$$
\begin{aligned}
fish(g)(i,j)_{move\_out} = & \\
& \left[\min\left(k_1 \times \sum_{\substack{water(i,j)> \\ water(x,y)}} \frac{water(i,j) - water(x,y)}{water(i,j)}, maxwater(g)\right) + \right. \\
& \left. \min\left(k_2 \times \sum_{\substack{fish(g)(i,j)> \\ fish(g)(x,y)}} \frac{fish(g)(i,j) - fish(g)(x,y)}{fish(g)(i,j)}, maxdensity(g)\right)\right] \\
& \times fish(g)(i,j)^n, \quad (5)
\end{aligned}
$$

where $fish(g)(i,j)$ is the total number of fish in functional group g in cell $C_{i,j}$, $water(i,j)$ is the water depth of cell $C_{i,j}$, $maxdensity(g)$ is the maximum fraction of fish in functional group g that can move due to density differences, and $maxwater(g)$ is the maximum fraction of fish that can move due to water differences. $C_{x,y}$ is one of the landscape cells adjacent to cell $C_{i,j}$. $fish(g)(x,y)$ is the total number of fish in functional group g in cell $C_{x,y}$, and k_1 and k_2

are empirically-based parameters. The sums in Eq. (5) are taken over the eight neighboring cells that satisfy the given inequality. The first sum in Eq. (5) is taken over all adjacent cells that have higher water depths than the cell being evaluated. The second sum in Eq. (5) is taken over all adjacent cells that have lower fish densities than the cell being evaluated. Please noted that the amount of fish move into cell $c_(i, j)$ is not explicit calculated, but implied computed by applying Eqs. (5) on its eight neighboring cells.

As the previous section, we use the same simplified scenario to demonstrate diffusive fish movement. After *density-independent fish movement*, only the small and medium size fish exist in the central cell, $C_{i,j}$. Those fish have different movement patterns according to the difference of water depth and fish densities. The small size fish can move to one of the five adjacent cells in light/medium/dark gray. Thin arrows in the right plot of Fig. 5 represent those movements. Due to the limits of water depth, the medium size fish in the central cell can only move to one of two adjacent cells in medium/dark gray. Those movements are represented by bold arrows in the right plot of Fig. 5. Similarly, the small size fish in cell $C_{i-1,j-1}$ can move to the adjacent cells according to the relative difference of density and water depth. Bold arrows in the right plot of Fig. 5 represent those movements. This type of movement does depend on fish densities, so the changes of fish density must be stored until all calculations are completed, and then added to the original landscape matrix to create a new fish landscape matrix for the next timestep.

2.2.3 Fish Mortality, Aging, Growth, and Reproduction

In ALFISH, fish mortalities depend on age and prey availability. Food-based mortality is calculated as the ratio of prey biomass in the cell to the amount needed for maintenance times a function giving the amount of available prey in the cell. Age mortality is directly related to fish size and age. Food-based mortality is compared to age-dependent mortality for each age class, and the greater of the two is applied. Fish that survive through the mortality phase will grow and age. The age classes for the fish functional groups are 30-days. If the timestep is at the end of a 30-day period, all of the fish are moved to the next age class. All age classes are divided into six size classes, but all fish in an age class are within a single size class at any timestep. All fish within an age class advance synchronously in size class. Therefore, size class, (or $sizeClass$) can be stored as a single integer ranging from 1 to 6 representing which size class each age class is in. This integer is incremented by one each timestep (5 days), and then if the integer is greater than 6 (all in this age class automatically advances to the next age class), it is reset to 1. For each functional group, if it is the appropriate time of year, the number of offspring is calculated using 0.5 times the number of fish of reproductive age for that functional group times the number of offspring per female per reproductive event. To prevent the population from producing too many new fish in a reproductive event, a constant maximum reproduction density is used.

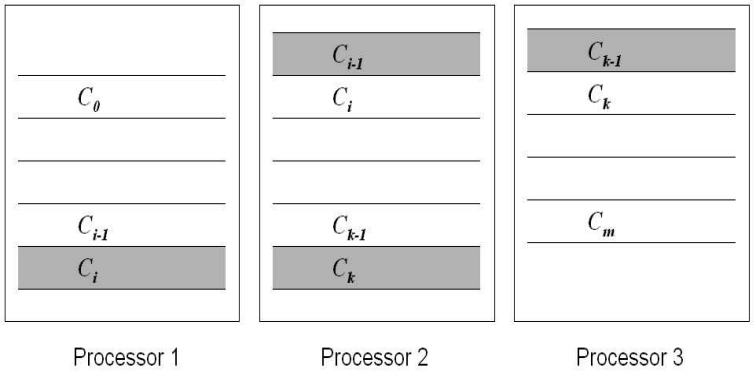

Figure 6: Landscape partition on 3 processors

3 Parallel Methodology

To parallize ALFISH, several modifications to the sequential model were required. A two-layer communication model was deployed to provide message-passing functions between all processors. Considering that sequential output operations are needed in each timestep, one processor (P_0) was dedicated to collect fish data from other processors, referred to as computational processors, and store those data. Each computational processor only simulates fish behaviors in a row-size, block-striped partition of the landscape.

3.1 Fundamentals of Landscape Partitioning

In the parallel ALFISH model, a *region area matrix* is used to remove excluded areas from the landscape map, and this matrix is duplicated on all processors in order to reduce data immigration time. The entire landscape map is statically partitioned among all computational processors. In order to enable inter-processor data communication between adjacent processors, a ghost row is attached to the upper and lower side of each partitioned landscape. Fig. 6 shows a partition of a landscape map on three computational processors. One part of the landscape map, an internal region (row C_0 to row C_{i-1}) plus a ghostzone (row C_i), is assigned to processor 1. Another part of the landscape map, an internal region (row C_i to row C_{k-1}) plus two ghostzones (row C_{i-1} and row C_k), are assigned to processor 2. The remaining part of the landscape map, an internal region (row C_k to row C_m) plus a ghostzone (row C_{k-1}), is assigned to processor 3. Ghostzones in Fig. 6 are represented in gray, and used to mimic fish movements as they appear in the sequential code.

3.2 Computational Intensity of Landscape Cells

In order to improve model performance, computational intensity (CI) (defined as CPU time used to process data in a landscape cell) was introduced. From previous efforts [11, 12, 13], the computational intensity of a landscape cell is attributed to a number of factors (e.g., current hydrology data, fish structure, and so on). It is neither trivial nor currently feasible to determine the real time computational intensity, therefore two methods was applied. In the first method, we assign the averaged computational intensity to each landscape cell, which is defined by

$$CI(i,j) = \frac{W}{S}, \tag{6}$$

where $CI(i,j)$ is the computational intensity of cell $C_{i,j}$, W is the total workload at each timestep, and S is the total number of landscape cells. This method results in static landscape partitioning, and the evaluation data from model performance can provide basic information on processor load imbalances introduced by the dynamic change of environmental data. Furthermore, it is a base scenario, against which we can further evaluate the performance of the parallel ALFISH model with dynamic load-balancing.

To take count of the uneven workload distribution among processors during simulation, in the second method, the average computational intensity of all landscape cells on each computational processor was used to evaluate the computational intensity of each cell. Therefore, Eq. (6) becomes

$$CI(i,j) = \frac{W_p}{S_p}, \tag{7}$$

where $CI(i,j)$ is the average computational intensity of a landscape cell on processor p, W_p is the workload on processor p, and S_p is the number of landscape cells on processor p. Based on Eq. (6) or Eq. (7), the computation intensity of a landscape row is calculated as the sum of the computation intensity of each cell in the row, that is

$$CI_i = \sum_{j=1}^{n} C(i,j), \tag{8}$$

where n is the number of cells in the landscape row, and i is the row number.

3.3 Dynamic Load-Balancing and Imbalance Tolerance

Dynamic load balancing techniques are now widely used in scientific computation [14, 15] where static partitioning fails to equally distribute the load among different processors. Our procedure for load balancing consists of the following operations:

3.3.1 Determine Workload

Each computational processor determines its workload (W_p) at each timestep by

$$W_p \equiv T_{comp} = T_{wall} - T_{overhead} \tag{9}$$

where W_p is the workload on processor p, T_{comp} is the time to process the local data on processor p, T_{wall} is the wall-clock time elapsed from the previous timestep, and $T_{overhead}$ is the time used for packaging and exchanging data plus waiting time at this timestep. Each processor also determines the CI of its local landscape cells, and sends the workload information including the local cells' computational intensities and size of the landscape partition to the master processor.

3.3.2 Workload Redistribution

The master processor compares all workloads on computational processors and determines which landscape row(s) need to be redistributed, if an imbalance is found. The following equations were used to determine the average workload, and maximum workload imbalance:

$$\begin{aligned} MIB &= \max_{1 \le p \le nps} \left(\frac{|W_p - W_{avg}|}{W_{avg}} \right) \times 100, \\ W_{avg} &= \frac{1}{nps} \sum_{p=1}^{nps} W_p, \end{aligned} \tag{10}$$

where W_{avg} is the average workload at the previous timestep, W_p is the workload on processor p, and nps is the number of computational processors. MIB is the maximum imbalance index. The dynamic load-balancing phase itself brings extra overhead to the simulation and results in significant data movement among processors. Therefore, an appropriate imbalance tolerance must be determined in the simulation to avoid excessive and perhaps unnecessary load-balancing. The function for landscape repartitioning is shown in Fig. 7.

3.3.3 Data Immigration

After receiving new partition information, each importing processor ($W_p < W_{avg}$) notifies the exporting processor ($W_p > W_{avg}$), and receives those landscape row(s) determined by the master processor. Synchronization is also required to guarantee that all processors enter the dynamic load-balancing phase simultaneously.

3.4 Parallel Computational Model

In the parallel ALFISH model, explicit message-passing is used to mimic the same fish dynamics described in the Section 2.2. The computational model is shown in Fig. 8. Compared to the flow chart shown in Fig. 4, the parallel

```
void repartition(W_p, nps, S_p, partition, imbalance_tolerance)
{ compute W_avg and MIB;
  compute CI_i on each processor using W_p;
  if (MIB > imbalance_tolerance) {
     rank=1;
     partition[0]=0;
      loads =0;
      for i =1 to max_num_of_row − 1 {
          loads+=CI_i ;
          if ( loads <= W_avg) and (loads + CI_{i+1} > W_avg) {
             partition[rank] = i;
             rank++;
             loads=0;
          }
      }
      partition[nps] = max_num_of_row;
  }
}
```

Figure 7: Landscape repartition function

ALFISH model contains explicit data exchange and computational synchronizations (represented by dashed lines) and a computational phase to calculate computational intensity and redistribute cells. As shown in Fig. 8, within each timestep four explicit synchronizations are required. The synchronization before the phase *Compute CI and redistribute cells* guarantees that all processors enter the dynamic load-balancing phase at the same time. Since the movement of fish depends and has influence on the adjacent cells, an explicit data exchange is required after the computation of *Density-Independent Fish Movement* and after the computation of *Diffusive Fish Movement*.

In the parallel ALFISH model, blocking LAM-MPI primitives (i.e., MPI_Send, MPI_Recv and MPI_Sendrecv) are used. The main reason for choosing blocking communication primitives is that such primitives eliminate the code segment for explicit synchronizations, which are required several times within each timestep. In addition, MPI_Barrier is used to guarantee synchronization during the simulation.

3.5 Parallel Density-Independent Fish Movement

Since this fish movement is density-independent, the order of computation is not an influential factor for the simulation. In the parallel ALFISH model, the internal regions of a landscape partition (on each processor) are treated in the same manner as those in the sequential model. In order to enable the same computations as in the serial model, partial results must be calculated

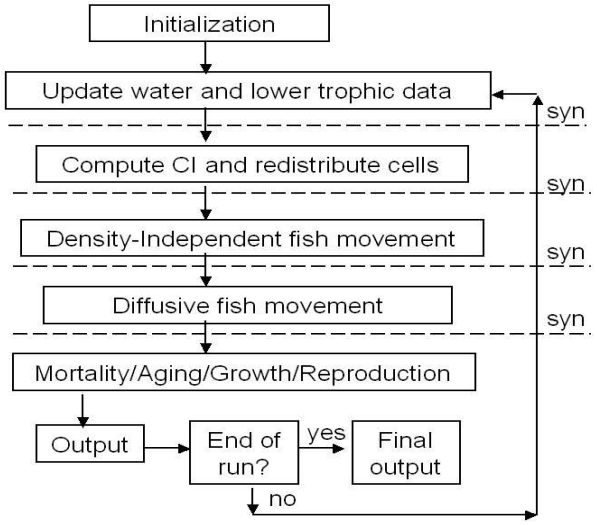

Figure 8: Computational model for parallelizing ALFISH

simultaneously, and data exchanges are required between adjacent processors. For more detailed information, please refer to [12].

3.6 Parallel Diffusive Fish Movement

Similar to the calculations of *density-independent fish movement*, each processor only simulates the fish movements in internal cells. The calculations of diffusive fish movement are similar to those described in Section 3.5, with the exception that this particular movement depends on fish densities. Therefore, the changes of fish density in internal cells are temporarily held in storage until all calculations are complete, and then added to the original landscape matrix to create a new fish landscape matrix for the next calculation. At each timestep, the changes of fish densities in the ghostzone are stored. After the calculation of fish movements on each processor, the changed values of fish densities in a ghostzone and the fish density on its adjacent row are passed to the adjacent processor to mimic the computations in the sequential ALFISH model.

3.7 Parallel Fish Mortality, Aging, Reproduction, and Growth

The calculations of fish mortality, aging, reproduction and growth depend on local information about the cells where fish exist. This local information includes the values of lower trophic data, age status of fish, and so on. In the parallel ALFISH model, these computations are basically the same as in the sequential model and are only executed on cells in the internal region of the landscape map (on each processor).

4 Verification and Performance

In this section, a standard hydrology scenario was applied to verify the accuracy and the performance of the parallel ALFISH model.

4.1 Scenarios

The ALFISH models are mainly used to determine the pattern of fish density on the landscape for a variety of hydrology scenarios. The motivation for the particular scenarios chosen was the Restudy process [16] for the selection of a plan for Everglades restoration. One particular scenario used, F2050, is a standard base scenario which uses water data from a 31-year time series of historical rainfall from 1965 through 1995, as well as sea level, population level and socioeconomic conditions projected for the year 2050. It also includes all of the previously legislated structural changes for water control measures. Therefore, the simulation time of both the sequential and parallel ALFISH model is typically 31 years and the timestep is 5 days.

4.2 Comparison of Selected Outputs

To verify the parallel model's *correctness* and accuracy, that is, its ability to produce results similar to those of the sequential model, we compared outputs of both the sequential model and parallel model. We produced and analyzed several outputs, and chose two sets of data for comparison. One is a time series of mean fish density across the entire landscape, and the other is a 31-year mean fish density and distribution on April 1. Fig. 9 illustrates the comparison of a time series of mean fish density across the entire landscape. The left graph represents the output from the parallel ALFISH model, the right graph is the output from the sequential ALFISH model. The results show little difference in total fish densities when averaged over the entire region in the entire 31-year time period, the maximum difference is less than 0.01%. Fig. 10 shows mean fish density map comparisons on April 1. Again, there are no meaningful differences between the outputs of the parallel and sequential ALFISH models. The above comparisons are averaged over space or time. Comparisons have also been made of maps similar to Fig. 10 for single dates, with no meaningful differences found in these either.

4.3 Parallel Performance Analysis

In order to further study the parallel model performance, we recorded the actual computational time and overall overhead time on each processor. The overall overhead time includes the time for data communication, the time used for data message packing and unpacking, as well as the idle time due to synchronization.

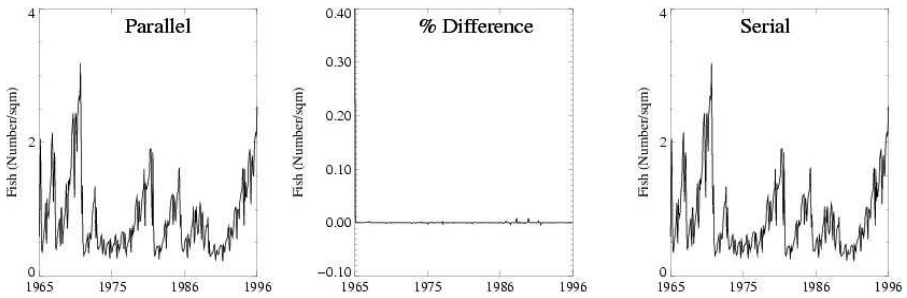

Figure 9: Time series for mean fish density in fish/m^2 as averaged across the entire landscape

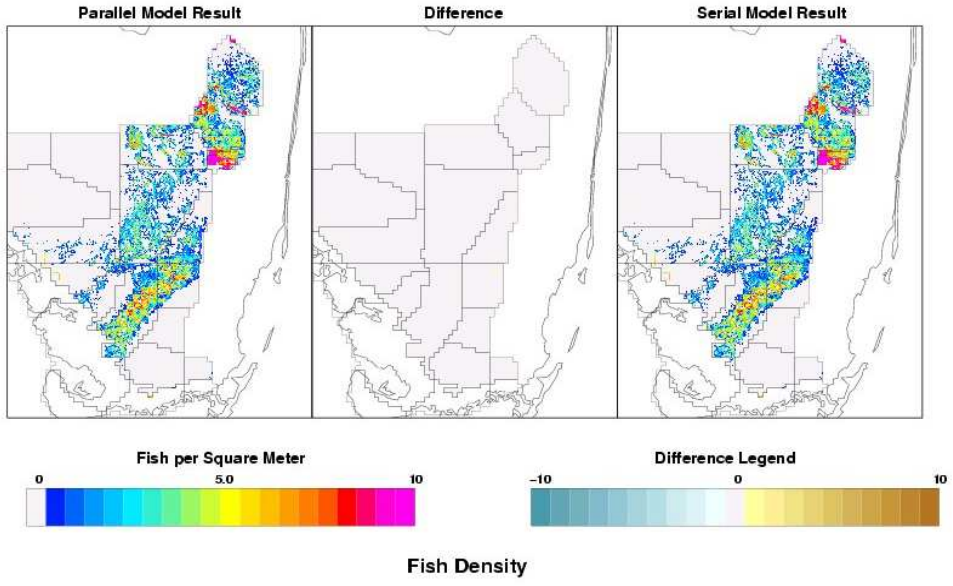

Figure 10: Spatial averaged fish density map comparison in Everglades on April 1, averaged over all years

Table 1: Model performance profile of the parallel ALFISH model with static partitioning (unit:second)

	Proc 0	Proc 1	Proc 2	Proc 3	Proc 4
Computation	1580	21969	28515	32109	25479
Overhead	32093	9252	3301	679	5765
Initialization	17	17	17	17	17
File (I/O)	N/A	612	606	617	609
Sum	33700	31850	32439	33422	31870

4.3.1 Static Load Balancing

The first method (see Eq. (6)) for calculating computational intensity results in a static partitioning that we would expect to produce an unbalanced workload across the computational processors. Table 1 contains a performance profile of a model run on 5 processors. From this profile, we make the following observations:

1. Most of the time, Processor 0 is waiting for the data from computational processors, and only 5% of the total execution time is used for data output;

2. Initialization time is not significant, which implies that it is reasonable to reduce the communication time by duplicating the landscape mask on each processor during initialization;

3. Although at each timestep the model must read water data from disk and update hydrology data, those file operations are not very time consuming, and only take 2% of total execution time;

4. The workload is not well balanced among all computational processors

On the most lightly loaded processor 1, only 63% of the execution time is used for computation, while the overall overhead consumes over 25% of the total execution time. On the most heavily loaded processor 3, computation takes almost 95% of the execution time, while the overall overhead reflects only 2% of the total execution time. In other words, on processor 3, the time for data communication, message packing and unpacking requires only 2% of the total execution time (since idle time can be ignored from the overall overhead). At each timestep, computation time (T_{comp}) consists of three parts: time for computing *density-independent fish movement* (T_{intra}), time for computing *diffusive fish movement* (T_{move}) and time for computing other activities: *fish mortality, reproduction, aging and growth* (T_{other}). Fig. 11 illustrates different components of computation time (T_{comp}, including T_{intra}, T_{move}, T_{other}) on processor 3 over one simulation year. The horizontal axis is days in a year, and the vertical axis is computation time in seconds. The maximum and minimum computation times for *density-independent fish movement* at each time step (T_{intra}) are

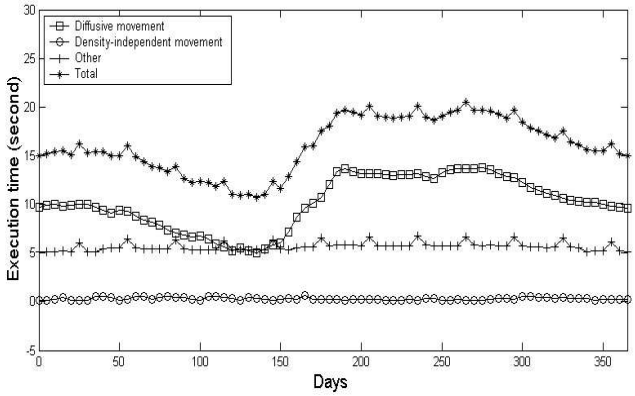

Figure 11: Computation time on processor 3 with static partitioning in a given year

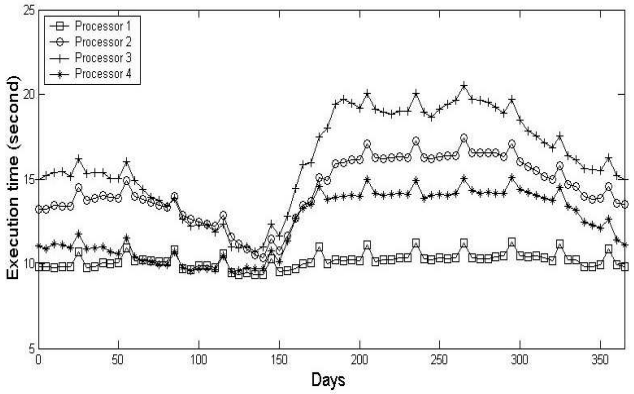

Figure 12: Computation time on 4 processors in a given year

about 1.2 and 0.2 seconds, respectively. Such minimal variance suggests that the assumption of constant computational intensity in this phase is valid. The maximum and minimal computational times for fish mortality and other behaviors are about 7 and 5 seconds, respectively, and at the last day of every 30 days, the fish growth and aging contribute just under 1 second to T_{other}. Since the variance of T_{other} is not significant relative to the average value of T_{other}, it is also reasonable to assume that the computational intensity of each cell in this phase is constant. In contrast, the variance of T_{move} (ranging from 5 to 14 seconds) clearly suggests this particular component has the greatest influence on the total computation time at each timestep. Fig. 12 illustrates the differences of computation time on all 4 processors in the same given year, indicating that computational workloads are not well-balanced among the processors. For example, on day 200, computation time T_{comp} is about 20 seconds on processor 3, and only 10 seconds on processor 1. On day 150, the computation time on all processors is about 12 seconds. The computation time on processor 3 varies greatly according to the season of the year, while the computation time on processor 1 is about 10 seconds year round.

4.3.2 Dynamic Load Balancing

The second method (see Eq. (7)) is an improvement of the first method, which allows the model to better distribute the workload across all computational processors. Dynamic load balancing is achieved by evaluating the computational time on each processor at the previous timestep. If a serious imbalance is detected, the model will repartition the landscape matrix and redistribute the workload on all computational processors for the next calculation (i.e.,next timestep). A performance profile of the parallel ALFISH model with dynamic load balancing is in Table 2. Comparing Tables 1 and 2, the following observations can be made:

1. the simulation time of the parallel model using dynamic load balancing is 4 hours less than that of the parallel model using static partitioning;

2. the workload is better balanced among all computational processors.

Table 2: Model performance profile of the parallel ALFISH model with dynamic load balancing (imbalance tolerance = 20%, unit:second)

	Proc 0	Proc 1	Proc 2	Proc 3	Proc 4
Computation	1514	15848	16238	16283	16650
Overhead	18801	3186	2675	2553	2894
Initialization	18	18	18	18	18
File (I/O)	N/A	612	606	617	609
Sum	20333	19664	19637	19471	20171

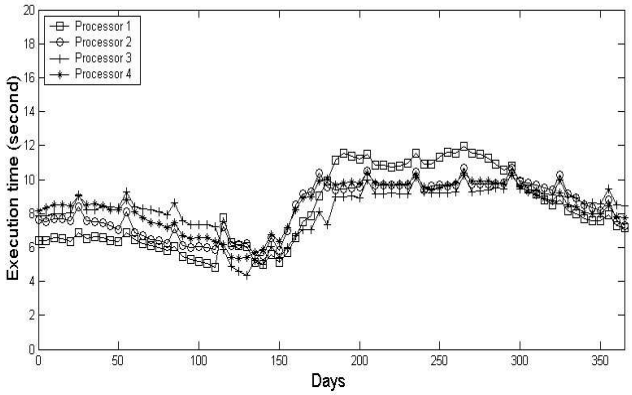

Figure 13: Computation time on 4 processors with dynamic load balancing in a given year

On the most lightly loaded processor 1, 78% of the execution time is used for computation, and the overall overhead consumes over 16% of the execution time. On the most heavily loaded processor 4, computation takes almost 82% of the execution time, while the overall overhead consumes 14% of the total execution time. Fig. 13 shows the computation time on all those 4 processors in the same given year, indicating the balanced computational workload across the processors. Fig. 14 shows the size of the landscape partition on those 4

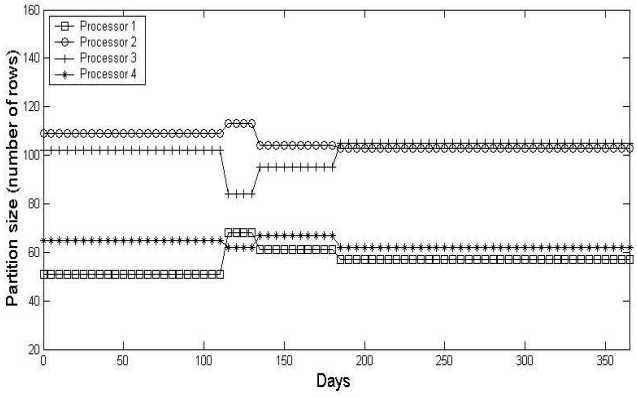

Figure 14: Partition size on 4 processors in a given year

processors during the one-year simulation. In our simulation, the imbalance tolerance (set to 20%) is used to trigger the repartition of the landscape matrix.

At each timestep, the master processor evaluates the workload distribution on each computational processor. If the imbalance is less than 20%, the model keeps the previous partition. If the imbalance is larger than 20%, the model repartitions the landscape matrix. This results in a change of partition size on days 115, 135, and 185 in Fig. 14 .

4.3.3 Performance Improvement for Parallel Model

In order to measure the performance gain of the parallel ALFISH model, we executed both the sequential and parallel implementations on the Sun Enterprise 4500 (see Section 1.3). We first ran the sequential model and recorded the execution time, followed by a series of parallel simulations using different numbers of computational processors (ranging from 3 to 13)[2]. Fig. 15 illustrates the execution time of the parallel ALFISH model. We note that even with the static partitioning of the landscape matrix, the parallel ALFISH model demonstrates remarkable speedup. The average execution time of the sequential model is approximately 35 hours. The execution time of the parallel ALFISH model with 13 computational processors is about 4 hours (the performance improvement factor being about 9). Using the same number of processors, the performance of the parallel model with dynamic load-balancing (imbalance tolerance = 20%) is even more impressive: the model turnaround time is less than 3 hours with a performance improvement factor approaching 12.

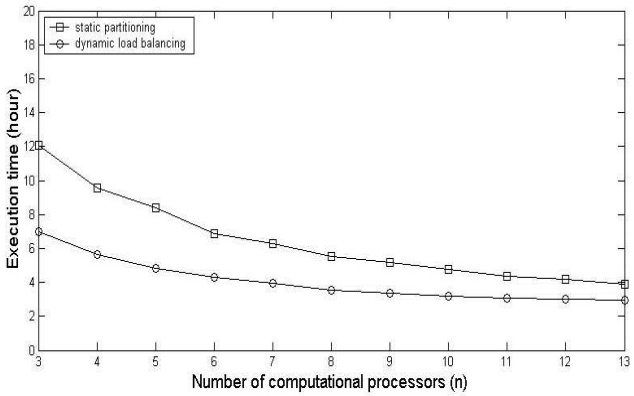

Figure 15: Total execution time of the parallel ALFISH model. (The averaged execution time for serial ALFISH model is around 35 hours)

[2]One processor is always dedicated to output operations not simulation. The capacity of this processor is reserved for future communications with other ATLSS models

5 Conclusion and Future Developments

The nearly identical outputs and excellent speed improvement obtained from the parallel ALFISH model, as compared with the sequential model, provide strong evidence that grid-based partitioning can be highly effective for age- and size-structured spatially explicit landscape fish models. Our results indicated that even using simple static partitioning, the parallel ALFISH model demonstrates remarkable scalability. Comparing the averaged execution time of the sequential model, which is about 35 hours, the parallel ALFISH model (using 13 computational processors) requires less than 4 hours. The performance of the parallel model with dynamic load-balancing on 13 computational processors is exceptional: adapting a dynamic load balancing method (imbalance tolerance = 20%), the model turnaround time is less than 3 hours yielding a speedup factor near 12. In this paper, one-dimensional partitioning (row-size, block-strip partition) was adapted, we are now developing dynamic load balancing techniques for two-dimensional partitioning (both row-size and column-size partition) using tiling algorithms [14, 17]. Using message-passing with the parallel ALFISH model, explicit data synchronization and data communication consume significant time, due to the complex age- and size-structure of the fish component. This indicates that the ALFISH model maybe more suitable for parallel execution on a shared-memory multiprocessors system using multi-threads. Further plans for the ALFISH model development include adapting finer grid resolution, taking account of the effects of complicated landscape features such as canals, and coupling with other ATLSS models for parallel multi-scale ecosystem simulation across heterogeneous components of a computing grid.

Acknowledgments

This research has been supported by the National Science Foundation under grant No. DEB-0219269. This research used the resources of the Scalable Intracampus Research Grid (SInRG) Project at the University of Tennessee, supported by the National Science Foundation CISE Research Infrastructure Award EIA-9972889. The sequential implementation was developed with support from the U.S. Geological Survey, through Cooperative Agreement No. 1445-CA09-95-0094 with the University of Tennessee, and the Department of Interior's Critical Ecosystem Studies Initiative.

References

[1] D. DeAngelis, S. Bellmund, W. Mooij, M. Nott, E. Comiskey, L. Gross, M. Huston, and W. Wolff, *Modeling Ecosystem and Population Dynamics on the South Florida Hydroscape*, In The Everglades, Florida Bay, and Coral Reefs of the Florida Keys: An Ecosystem Sourcebook, J. Porter and K. Porter, eds., CRC Press, (2002), pp. 239–258.

[2] D. DeAngelis, L.Gross, M. Huston, W. Wolff, D. Fleming, E. Comkskey, S. Sylvester, *Landscape modeling for Everglades ecosystem restoration*, Ecosystem, 1 (1998), pp. 64–75.

[3] L. Gross and D. DeAngelis, *Multimodeling: New Approaches for Linking Ecological Models*, in Predicting Species Occurances: Issues of Accuarcy and Scale, J. M. Scott, P.J. Heglund, M.L. Morrison, eds. (2002), pp. 471–476.

[4] H.Gaff, D. DeAngelis, L. Gross, R. Salinas, M. Shorrosh, *A Dynamic Landscape Model for Fish in the Everglades and its Application to Restoration*, Ecological Modeling, 127 (2000), pp. 33–53.

[5] W. F. Wolff, *An Individual-Oriented Model of a Wading Bird Nesting Colony*, Ecological Modelling, 114 (1994), pp. 72–75.

[6] D. Wang, E. Carr, M. Palmer, M. W. Berry, L. J. Gross, *A Grid Service Module for Natural Resource Managers*, IEEE Internet Computing, 9:1 (2005), pp. 35-41.

[7] *SInRG: Scalable Intracampus Research Grid*, Innovative Computing Laboratory at the University of Tennessee, Knoxville, TN, 2002. http://icl.cs.utk.edu/sinrg/.

[8] *LAM/MPI: Parallel Computing*, Indiana University, Bloomington, IN, 2002. http://www.lam-mpi.org/

[9] R.L.Fennema, C.T. Neidrauer, R. A. Johnson, T.K.MacVicar and W.A. Perkins, *A Computer Model to Simulate Everglades Hydrology*, In Everglades: The Ecosystem and Its Restoration, S.M.Davis and J.C.Ogden, eds., St. Lucie Press, (1994), pp.249–289.

[10] S. Duke-Sylvester and L. J. Gross, *Integrating spatial data into an agent-based modeling system: ideas and lessons from the development of the Across Trophic Level System Simulation (ATLSS)*, Chapter 6 in: Integrating Geographic Information Systems and Agent-Based Modeling Techniques for Stimulating Social and Ecological Processes, H. R. Gimblett, ed., Oxford University Press, (2002).

[11] H. Gaff, J. Chich, J. Trexler, D. DeAngelis, L. Gross, R. Salinas, *Evaluation of and insights from ALFISH: a spatial-explicit, landscape-level simulation of fish populations in the Everglades*, Hydrobiologia, 520 (2004), pp73-87.

[12] D. Wang, L, J. Gross, E. Carr, M. W. Berry, *Design and Implementation of dynamic fish model in South Florida on parallel architecture*, accepted by 37th Hawaii International Conference on System Sciences, 2004

[13] A. Immanuel, M. W. Berry, L. J. Gross, M. Palmer, D. Wang, *A parallel Implementation of ALFISH: Compartmentalization Effects on Fish Dynamics in the Florida Everglades*, Simulation Practice and Theory, 13:1 (2005), pp.55-76.

[14] K. D. Devin, J. E. Flaherty, *Parallel Adaptive hp-refinement Techniques for Conservation Laws*, Applied Numerical Mathematics, 20 (1996), pp. 367–386

[15] J. D. Teresco, M.W. Beall, J.E. Flaherty, M.S. Shephard, *A hierarchical Partition Model for Adaptive Finite Element Computation*, Computer Methods in Applied Mechanics and Engineering, 184 (2000) pp. 269–285.

[16] U.S. Army Corps of Engineers, *C&SF Restudy Draft Feasibility Report*, http://www.restudy.org/overview.pdf

[17] E.Leiss and H. Reddy, *Distributed load balancing: design and performance analysis*, W.M Keck Research Computational Laboratory, (1989) pp. 205–270