A Cellular Model of Fish Migration

Modeling and simulation

Dan Liu
Faculty of Engineering
University of Ottawa
Ottawa, Canada
dliu049@uottawa.ca

Abstract— **Normally, lots of dams are built over rivers which are used for irrigation or hydropower, but also become barrier for fish migration. In order to protect the ecosystem balance, some measures need be taken to facilitate the migration of fish. However, the budget which is devoted to improve the facility on river is always limited. So we need to build a model to simulate fish migration under different conditions so that different construction strategies imposed on river could be evaluated. In this paper a cellular fish migration is introduced and the modeling and simulation of model are also presented. Some analysis and discussion on simulation result are performed and finally a conclusion is given.**

Keywords— Model, Simulation, Cellular, Cell-DEVS, River, Fish, Migration.

I. INTRODUCTION

River plays an important role in our daily life which provides drinking water, transportation path and farming resource. Along with the exploitation of river, more and more facilities are built over river in order to make best use of river. For instance, there are always lots of dams built over river to exploit hydropower of river and control irrigation of river.

However, the increasing dams also become barriers for fish migration in river. Because of barriers of dams, the fish migration becomes more difficult, which means the speed of fish migration becomes slower and the rate of success fish migration becomes lower. A portion of fish maybe fail to pass the dam to migrate ahead, or die during the migration. It is not only a matter of success rate of fish migration, but also a matter of ecosystem balance. Not enough fish migrating to river source successfully will lead to the decrease of fish number born in the next year and consequently affect the normal propagation of various species living in the river. Finally, it may produce great damage to the entail ecosystem balance.

Considering the positive effect of dams on the ecosystem balance, some measures need to be performed to mitigate the positive effect of dams on the ecosystem balance and improve the success rate of fish migration so that the entail ecosystem balance would be protected. Normally, various construction will be built over river to help fish migration. For instance, a smooth pass built over dams specific to fish will improve the success rate of fish migration greatly.

The question is that the budget of construction is limited and how to make best use of it to help fish migration best. The river is always very long with lots of dams. Considering the limited budget, it is impossible to build fish migration facility for all dams. We can only select a part of dams to devote major budget to build best fish migration facility for them while devote little or no budget for other dams. How to select the dams which will be devoted major budget is crucial in the procedure. Obviously, it is not a simple problem. We can not pick up some dams randomly or just pick up the first ten dams or twenty dams. The key is that we need to know the effect on fish migration if a set of dams are selected to devoted major budget, and the difference of effect among strategies. Comparing the different effect of strategies, we are able to select the best strategy and decide how to devote the limited budget.

The motivation of this term project is just to provide a solution to the above issue, evaluating effect of different construction strategies on fish migration. Basically, I use Cell-DEVS to build a cellular fish migration model which simulates the procedure of fish migration and perform simulation under different conditions in order to compare effect of different construction strategies and help making decision on budget arrangement.

In this paper, firstly, some background and related work on fish migration will introduced. And because this term project is an improvement of my assignment two, my work of assignment two will be presented before I show my improvement work in my term project. In the part of my improvement on term project, first of all, because this term project is completed using the latest version of CD++, the latest version of CD++ as well as its' new features will be introduced and some comparisons on execution performance between new version and old version CD++ will be performed. Afterwards, the improvement in my term project will be introduced and discussed. A set of different models will be presented and compared. Simulation are performed on different models and corresponding results will be compared and analyzed. Finally, a conclusion is provided based on the discussion above.

II. BACKGROUND

A. DEVS and Cell-DEVS

DEVS is the base of Cell-DEVS, which is an increasingly accepted framework for understanding and supporting the activities of modeling and simulation[1]. DEVS is a sound formal framework based on generic dynamic systems, including well defined coupling of components, hierarchical, modular construction, support for discrete event approximation of continuous systems and support for repository reuse. DEVS theory provides a rigorous methodology for representing models, and it does present an abstract way of thinking about the world with independence of the simulation mechanisms, underlying hardware and middleware[1].

Generally, Cell-DEVS is an extension of DEVS which is based on the DEVS formalism and builds discrete-events cell spaces to simulate behaviors of model in cell space domain. The goal of Cell-DEVS is to build discrete-events cell spaces, improving their definition by making the timing specification more expensive[1].

Basically, Cell-DEVS can be regarded as the combination of Cellular Automata(CA) and DEVS, allowing the implementation of cellular models with timing delays. Cell-DEVS improves execution performance of cellular models by using a discrete-event approach. It also enhances the cell's timing definition by making it more expressive. Each cell is defined as a DEVS atomic model, and it can be later integrated to a coupled model representing the cell space. A coupled Cell-DEVS is composed of an array of atomic cells, with given size and dimensions. There are also border cells which can have a different behavior due to their particular locations and result in a non-uniform neighborhoods. Moreover, the model's couplings permit connecting these models with other external sub-models[1].

DEVS and Cell-DEVS were implemented in a modeling and simulation tool, called CD++ which is a modeling tool that was defined using the formal specifications of Cell-DEVS, and some basic simulation techniques[1]. The toolkit includes facilities to build DEVS and Cell-DEVS models. DEVS atomic models can be programmed and incorporated onto a class hierarchy programmed in C++[1]. Coupled and Cell-DEVS models are defined using a built-in language. Cell-DEVS coupled model specification includes the definition of the size and dimension of the cell space, the shape of the neighborhood and borders. The cell's local computing function is defined using a set of rules[1]. This toolkit was successfully used to develop different types of systems: biological (ecological models, electrical activity of the heart tissue, ant foraging systems, fire spread, etc.), physical (diffusion, binary solidification, heat transfer, etc.), artificial (traffic problems, seeking devices, etc.), and others[1]. The models execute through the activation of an abstract simulation engine that is completely independent from the models themselves. Consequently, we were able to develop different kinds of simulators[1].

B. Fish Migration Model

As we know, usually, some kinds of fish species migrate during their life cycle between fresh and seawater. Generally, two groups are distinguished: anadromous fish hatch in freshwater, spend most of their life in the sea and return to freshwater to spawn. Common examples are salmon, smelt, shad, striped bass, and sturgeon. The other group: catadromous species, like most eels, live in freshwater and spawn in seawater.

Nowadays, most of rivers are flow regulated by more and more dams. Dams built for shipping traffic, hydropower usage and flow regulation at most rivers all over the world hinder migration of these fish. They form a barrier for upstream migration, although often for example fish ladders are installed. And even though downstream passage is usually less problematic, especially if young small fish travel, populations suffer substantial loss. Consequently populations of migrating fish decrease. In another word, they form a barrier for migrating fish species and are one cause for sustainable disturbances of the ecosystem even to extinction of several fish species.

So one basic question is how to distribute financial resources to improve fish passage systems to obtain a maximal effect. In [2], a concept to model up and downstream fish migration with an individual based approach is presented, which allows us to test different strategies of resource distribution. The question arises how to invest the given financial resources yielding the best effect. To describe solutions a "permeability" coefficient is given to each dam. It characterizes how well fish can migrate. The value is interpreted as the probability that one fish migrates successfully over the respective dam in one try. A solution for the whole river can be formulated by a vector or tuple of such permeability coefficients.

According to [2], the fragmentation of rivers by dams suggests discrete units in space. Every cell will represent an impoundment which is the segment between two successive dams. A river without tributary rivers for example can be seen as a one-dimensional grid G. If we look only at upstream migration, in this simple case every cell will have only one neighbor cell, except the last cell at the river's source. To describe more complex river systems, like for example the river Moselle with its tributary system, we have to include branching. Then some cells will have two neighbors or even more although every cell is only neighbor of one other cell. This gives us a grid G of cells x with "branches" embedded in two-dimensional space. Every cell can take states from the set $E = 0,1,...,n$ where n is the total number of simulated individuals. The state of a cell is interpreted as the number of fish present. To every cell a number $Pi \in [0, 1]$ is assigned. We interpret it as probability that one fish successfully migrates into this impoundment in one try. For upstream migration it therefore describes the permeability of the dam located downstream of the cell. At confluences, where tributary rivers join, Pi can be seen also as the preference for fish to migrate into tributary rivers versus following the main stream. At these branching points cells have more than one neighbor. Consider for example a cell with two neighbors with

permeabilities P1 and P2. Suppose that fish try to migrate in one of the neighbors with equal probability. Then the probability that a fish successfully migrates into the first or second neighboring cell is P1/2 respectively P2/2.

Considering the problem of investing our resources and changing these permeabilities such that most fish will arrive in suitable spawning habitats, we may not specify Pi but the amount of resources Ri used on a certain dam. To every dam (or cell) we can give a function that gives Pi for any value of Ri. Usually this function will be nonlinear, i.e. if we want to double the permeability, the resources used at this dam have to be more than doubled. Suppose Pi0 be the actual estimated permeability of dam i. Then Pi = Pi0 + fi(Ri) where fi is the cost function of dam i.

[2] describes a conceptual fish migration model and provides a set of definition and formalism demonstrating the model very well.

Based on the concept proposed in [2], I complete my project , building the fish migration model, implementing the model with the latest CD++ and performing simulations.

III. PREVIOUS WORK

Since this term project is based on some previous work completed in my assignment two, firstly, I will introduce work completed in my assignment two.

Basically, this model is based on the concept proposed in [2]. In assignment two, I built a simple fish migration model with Cell-DEVS according to the concept proposed in [2] and implemented it with a old version CD++ which is installed in a Windows PC. Related simulations are also performed in Windows PC.

In my model, upstream migration will be considered only.

A. River Model

Normally, a river is supposed to be made up of fragments separated by dams. Each fragment is represented by one cell with a state (s ∈ E = {0, 1, …, n}, where n is the total number of simulated fish). The state of a cell is interpreted as the number of fish present in this cell. For each cell, a number pi ∈ [0, 1] is assigned, representing the probability that one fish successfully migrates into this cell in one try.

A river may have several branches which is also represented by cells. Fish migrates from the river entry to the source of mainstream or branches. Eventually, only a part of fish can arrive at the source successfully. We can obtain a total number of arriving fish after simulation which is used to assess the effect of selected strategy.

In my assignment two, a river is represented by coupled cells in a nowrapped 5 x 10 grid. Each cell is initialized by default value(-1 or 0). -1 represents the border of river while 0 represents fragments of river. I build a cellular river model as follow:

-1	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	0	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	0	0	0	0	0	0
-1	-1	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	-1	-1	-1	-1	-1	-1

Figure 1. River model

Just as shown above, the river has 2 branches. The most left 0 cell is river entry while the most right 0 cell is the source of mainstream. The top 0 cell is the source of up branch while the bottom 0 cell is the source of down branch.

Fish entries the river at the most left 0 cell and eventually arrives at the most right 0 cell, or top 0 cell or bottom 0 cell.

B. Neighbor Cell

For each cell, its' neighbors are defined as follow:

```
neighbors : migration(-1,-1) migration(-1,0) migration(-1,1)
neighbors : migration(0,-1) migration(0,0) migration(0,1)
neighbors : migration(1,-1) migration(1,0) migration(1,1)
```

Figure 2. Neighbor cell definition

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

Figure 3. Neighbor cells

C. Migration Model

At the beginning, a number of fish, says 500, entries the river which means the most left 0 cell will be set to 500. Every interval, a small part of fish, says 2 fishes, will set out to begin its migration which means the value of most left 0 cell will be reduced gradually. And only a portion of fishes can migrate into the next cell successfully while the other portion of fishes die or disappear. A number pi ∈ [0, 1] is assigned to every 0 cell which represents the probability that one fish successfully migrates into the next cell.

The next cell's value will be set to the number of survival fishes from last cell after one interval. During the next interval, the fishes in current cell will try to migrate to the next cell and survival fished from last cell will arrive again.

Normally, in this model, fish tries to migrate to their left cell every interval. However, if there is a branch, fish will decide to migrate to the next cell of mainstream or the branch.

In this model, we suppose the probabilities of those 2 cases are all 50%, which means, at the conjunct cell, fish migrate to the next cell of mainstream in the probability of 50%, or to the branch in the probability of 50%.

The migration procedure will be iterated until all fishes at the river entry complete their migration. At that time, all survival fish is distributed at the source of mainstream or branches. Each source cell contains a number of fish which complete migration successfully. The data can be used to analyze and evaluate effect of different strategies.

D. Simulation and Results

As described above, Pi is assigned to each cell of river which represents the probability that a fish can migrate to this cell successfully. 2 sets of Pi are used during simulation. One is a constant Pi = 0.57. The other one is a function of the distance between current cell and the river entry cell, which can be expressed as follow:

Pi = (9 - cellPos(1)) / 9) in cell of mainstream,

Pi = ((9 - cellPos(1) + 2 - cellPos(0)) / 9) in cell of up branch stream,

Pi = ((9 - cellPos(1) + cellPos(0) - 2) / 9) in cell of down branch stream.

Consequently, 2 simulations are performed according to different Pi set.

a) Migration_1

In this simulation, Pi is a constant 0.57.

500 fishes is placed into the river entry to start the simulation.

The simulation result is demonstrated as follow:

```
Line : 110 - Time: 00:00:00:000
          0       1       2       3       4       5       6       7       8       9
    +-------------------------------------------------------------------------------+
  0|  -1.00           -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
  1|  -1.00           -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
  2|                                                                       500.00|
  3|  -1.00   -1.00          -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
  4|  -1.00   -1.00          -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
    +-------------------------------------------------------------------------------+
```

Figure 4. Simulation result a of migration_1

```
Line : 14968 - Time: 00:00:25:900
          0       1       2       3       4       5       6       7       8       9
    +-------------------------------------------------------------------------------+
  0|  -1.00    0.45  -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
  1|  -1.00          -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
  2|   0.79                                                                        |
  3|  -1.00   -1.00          -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
  4|  -1.00   -1.00    1.59  -1.00   -1.00   -1.00   -1.00   -1.00   -1.00   -1.00|
    +-------------------------------------------------------------------------------+
```

Figure 5. Simulation result b of migration_1

Some screenshots of dynamic simulation procedure video are shown as follow:

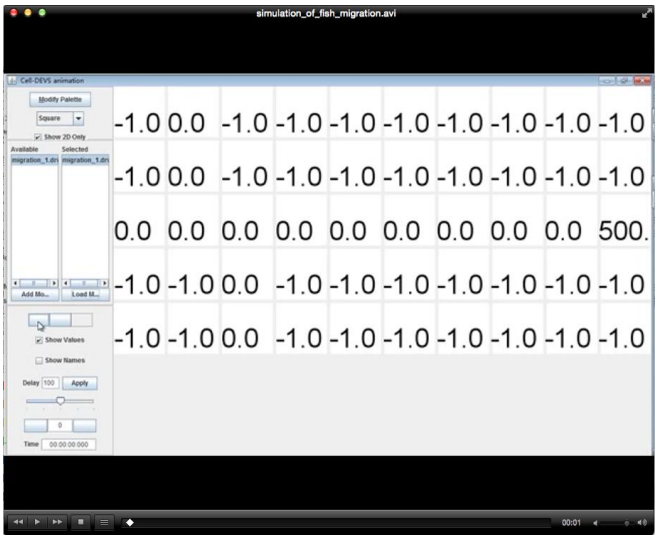

Figure 6. Beginning screenshot of dynamic simulation procedure video of migration_1

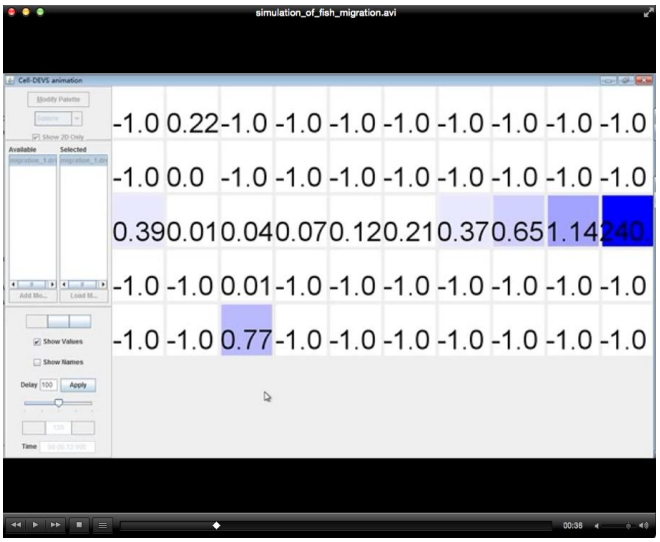

Figure 7. Execution screenshot of dynamic simulation procedure video of migration_1

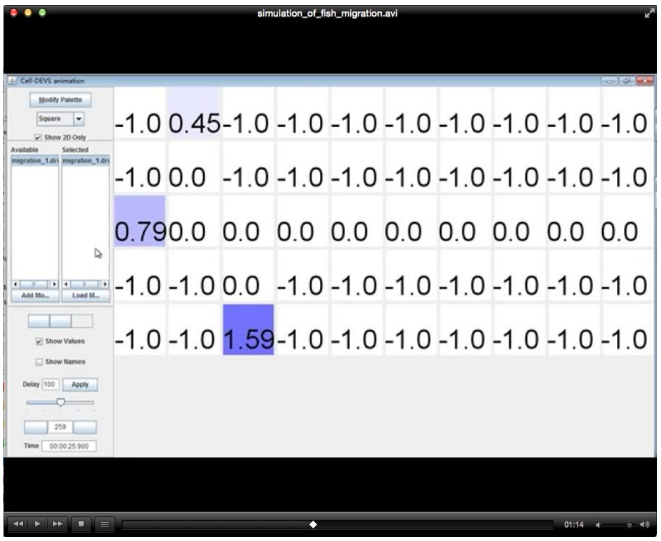

Figure 8. End sceenshot of dynamic simulation procedure video of migration_1

As shown, eventually, 0.79 fish arrive at the source of mainstream successfully while 0.45 fish at the source of up branch and 1.59 fish at the source of down branch.

b) Migration_2

In this simulation, Pi is a is a function of the distance between current cell and the river entry cell.

500 fishes is placed into the river entry to start the simulation.

The simulation result is demonstrated as follow:

```
Line : 110 - Time: 00:00:00:000
          0       1       2       3       4       5       6       7       8       9
   +-------------------------------------------------------------------------------+
 0|  -1.000          -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
 1|  -1.000          -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
 2|                                                                       500.000|
 3|  -1.000  -1.000          -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
 4|  -1.000  -1.000          -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
   +-------------------------------------------------------------------------------+
```

Figure 9. Simulation result a of migration_2

```
Line : 14968 - Time: 00:00:25:900
          0       1       2       3       4       5       6       7       8       9
   +-------------------------------------------------------------------------------+
 0|  -1.000   0.130  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
 1|  -1.000          -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
 2|   0.117                                                                       |
 3|  -1.000  -1.000          -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
 4|  -1.000  -1.000   0.234  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000  -1.000|
   +-------------------------------------------------------------------------------+
```

Figure 10. Simulation result b of migration_2

Some screenshots of dynamic simulation procedure video are also shown as follow:

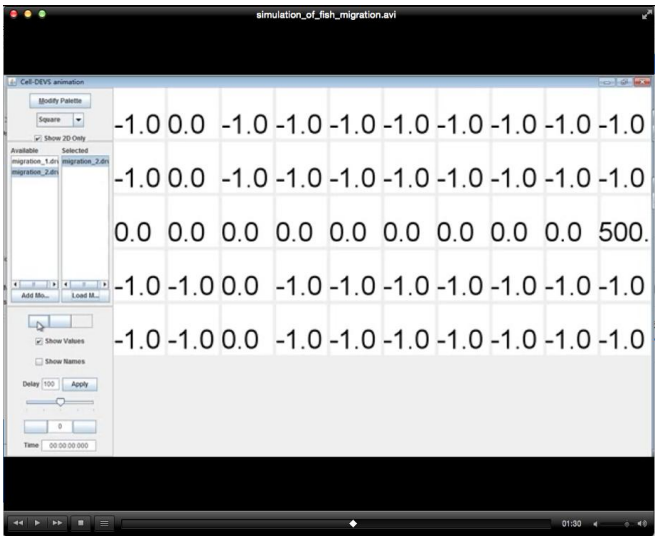

Figure 11. Beginning screenshot of dynamic simulation procedure video of migration_2

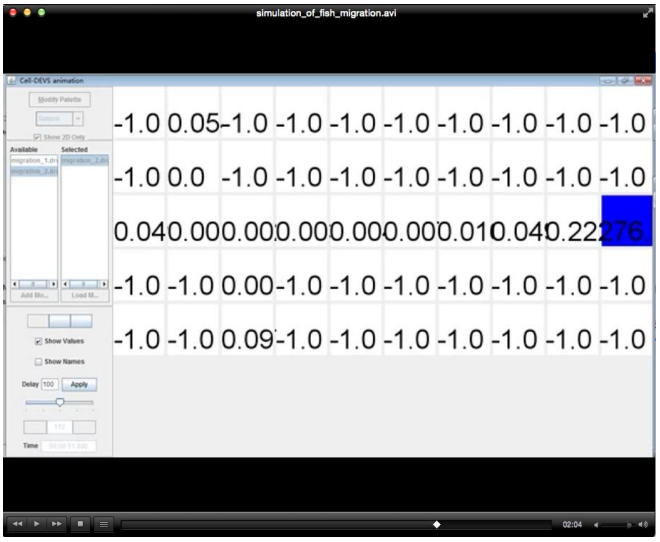

Figure 12. Execution screenshot of dynamic simulation procedure video of migration_2

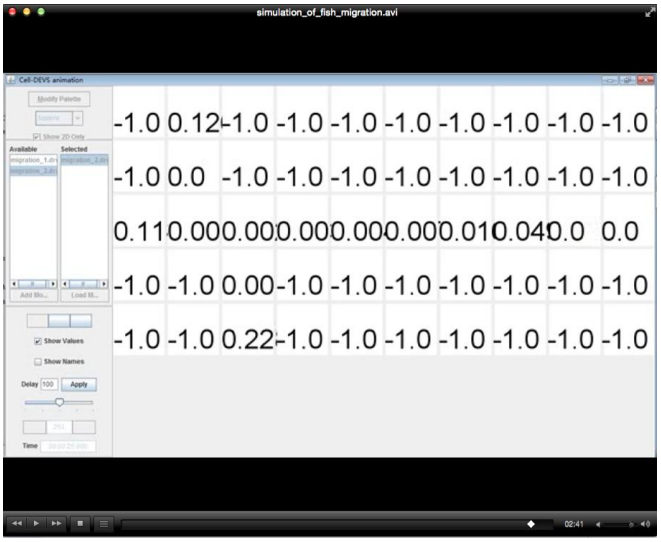

Figure 13. End screenshot of dynamic simulation procedure video of migration_2

As shown, eventually, 0.117 fish arrive at the source of mainstream successfully while 0.130 fish at the source of up branch and 0.234 fish at the source of down branch.

IV. NEW VERSION OF CD++

Since the improvement of assignment two is completed with the latest version of CD++ while assignment two is completed with an old version of CD++, the latest version of CD++ as well as its' new features will be introduced firstly.

The old version CD++ is a software installed in Windows PC and all of work is performed in local Windows PC. However, the latest version of CD++ has a totally different architecture which provides user a remote client in local PC and the simulation will be performed in remote server with the interface of local client[3]. The client provides user various functionality including creating model framework, uploading model files, starting simulation and fetching simulation results.

This novel architecture offers users more powerful computation capacity and much better performance of simulation.

However, a bigger improvement of new version of CD++ is the enhancement of build-in language ability of CD++ which make CD++ a much more powerful tool of modeling.

Briefly, there are two main enhancement of the build-in language: multiple state variables and multiple neighbor ports[3].

a) Multiple State Variables

In the old version of CD++, there is only one state variable for each cell which representing a unique state value of cell.

In the latest version of CD++, multiple state variables are supported. For each cell, multiple state variables can be defined and used . Each state variable can be initialized with different value and accessed respectively during simulation. This extension enhances the modeling ability of cells greatly.

The new version of CD++ also introduces a new file type: .stvalues file to support initializing multiple state variables at beginning.

The latest version of CD++ is also compatible to the unique state variable of cell in old version CD++ which means you can use the old unique state variable and the new multiple state variables simultaneously.

b) Multiple Neighbor Ports

Another major enhancement of the latest version of CD++ is the support of multiple neighbor ports.

In the latest version of CD++, it will be able to define multiple neighbor ports for each cell which means cells can exploit multiple neighbor ports to interact and communicate information which enhances the communication ability among cells greatly.

V. IMPROVEMENT

Based on the work completed in assignment two, I make a set of improvement to the existing fish migration model with the latest version of CD++ in order to enhance the functionality of the model.

Briefly, I summarize the improvement in my term project as follow:

1. Reproduce the generation of fish in the model which will generate and input migration fish into the original model in a varying frequency.

2. Considering budget devoted into improve dams is limited, I build a function of dam permeability(or probability of success migration from one cell to another) which can reflect different quantities of resource devoted into dams, and apply it to the migration model.

3. Take into account the spacial effect of cell on fish survival, the more crowed a cell is the lower probability fish survives, which will affect the final result of fish migration greatly.

4. Take into account the spacial effect of cell on migration speed, the more crowed a cell is the slower fish migrates, which will affect migration time.

5. Take into account the fitness of fish, for example, the longer fish migrates the bigger probability fish dies, and the more time fish tries to migrate from one cell to another the more exhausted fish become and dies when it's fitness fall below a value.

6. Build a more complex model for river which will contain larger grid, more branches and more complex combination of stream directions.

A. River Model

The new river model is based on the old one used in assignment two, however, more complex which contains larger grid, more branches and more complex combination of stream directions.

The new river model is shown as follow:

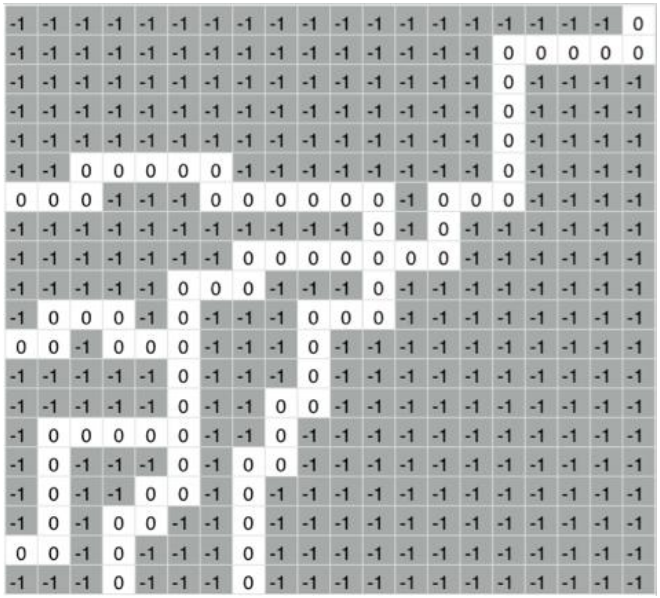

Figure 13. New river model

As shown above, the new river model has a 20*20 grid with more branches and more complex combination of stream directions. Even there is one crossing with 3 branches simultaneously.

Similarly, each cell has state variable representing the path of river: -1 for land and 0 for river fragment.

B. Neighbor Cell

With the new features of the latest version of CD++, it is able to model the migration with less neighbor cells.

For each cell, its' neighbors are defined as follow:

```
neighbors :                    migration(-1,0)
neighbors :  migration(0,-1) migration(0,0)  migration(0,1)
neighbors :                    migration(1,0)
```

Figure 14. Neighbor cell definition in new river model

C. Migration Model

The most important part of improvement is the new migration model.

With the power of the latest version of CD++, it is able to implement more functionality and more complex migration model. As summarized above, a set of improvement on migration model are completed with the latest version of CD++.

Basically, the migration model is defined with a set of migration rules. With different migration rules, the migration behaviors differently.

Firstly, several state variables are defined to achieve various goal of migration model as follow:

```
%from_v
%0:undefined, 1:up, 2:right, 3:down, 4:left
statevariables: path_v from_v number_v portion_v probability_v branch_v migrator_v display_v start_v initialize_v
statevalues: -1 0 0 0.8 0.9 1 0 0 0
initialvariablesvalue: migration_1.stvalues
```

Figure 15. Definition of state variables in new model

As shown above, I define 10 state variables totally.

path_v : This state variable of cell represents the path of river, -1 for land and 0 for river fragment;

from_v : This state variable of cell represents the source direction of this cell. The value of this state variable ranges among 0, 1, 2, 3 and 4. 0 means the source direction has not been defined. 1 means the source of this cell fragment is from above, 2 from right, 3 from below and 4 from left. Initially, the value of from_v is initialized with 0(undefined) and it will be set to the proper value according to the position of current cell during the beginning of simulation.

number_v : This state variable of cell represents the total number of fish inside cell currently. The value of number_v reflect the extent of crowding inside cell, which will vary dynamically during the simulation with some fish migrate in and some other out. What's more, the final value of cell located at the source of main stream and branches shows the number of fish migrating successfully.

portion_v : This state variable of cell represents the portion of total fish which will migrate in the next time slice. Since the space of river fragment is limited and the capacity of dam passage is limited, only a portion of fish are able to continue migrate each time which means the other should stay in the current cell waiting for the next time slice.

probability_v : This state variable of cell represents the probability of success migration for fish from current cell to next cell, which means only a part of migrating fish will arrive the next cell successfully while the other fail or die. Obviously, this state variable plays a crucial role in the migration model because it determine the number of fish migrating to the next cell successfully directly. The value of this state variable is affected by various conditions, for instance, the location of cell, the space of the cell and the resource devoted to improve facility in this cell. Considering various conditions affecting the value of probability_v, a function of probability_v is built to reflect changes of various conditions which will be presented later.

branch_v : This state variable of cell represents the number of branches in current cell. It is used to divide number of migrating fish at crossing cell. Normally, the value of this state variable will be 1, 2 or 3.

migrator_v : This state variable of cell represents the number of fish going to migrate to the next cell. A portion of them will arrive the next cell successfully.

display_v : This state variable of cell is used to control the display of current state value of cell. The CD++ only displays the unique state value of cell during simulation. However, the unique state value of cell and ports of cell can not be output simultaneously during one time slice. So the CD++ is controlled to output ports of cell when display_v = 0 and to display the state value of cell when display_v = 1. Consequently the dynamic simulation can be observed by watching the varying state value of cells.

start_v : This state of variable of cell is used to control the start of the entail simulation. Since at the very beginning of simulation, neighbor ports of cell are not initialized in the latest

CD++, which will fail the normal migration simulation relying on the values of neighbor ports, the start_v controls the first stimulation of neighbor ports. After this, the value of start_v will be set to 1 and then step into the normal simulation procedure.

initialize_v : This sate variable of cell is used to control the start of migration. Before the beginning of migration, some initializing work should be finished, for instance, the value of from_v, probability_v and branch_v should be initialized. After initialization, the value of initialize_v will be set to 1 and then the migration starts.

resource_v : This state variable represents the resource devoted to improve facility in this cell. Apparently, this value affects the value of probability_v greatly. So it is taken in to account when building the function of probability_v.

Besides, a couple of neighbor ports of cell are also defined to perform communication among cells.

```
neighborports: migrator_p from_p path_p
```

Figure 16. Definition of neighbor ports in new model

As shown above, three neighbor ports are defined totally.

migrator_p : This neighbor port of cell is used to transmit the number of fish going to migrate to the next cell. Normally, the value of this port equals to migrator_v.

from_p : This neighbor ports of cell has the same meaning as from_v. Since the value of from_v need to be determined by the value of from_v of neighbor cells sometimes, this neighbor port is used to transmit the value of from_v among neighbor cells. Apparently, the value of from_p equals to the value of from_v.

path_p : This neighbor port of cell has the same meaning as path_v. Since cell need the value of path_v of neighbor cells to determine some initial value of state variables, for instance, branch_v, this neighbor port is used to transmit the value of path_v among neighbor cells. Apparently, the value of path_p equals to the value of path_p.

Based on state variables and neighbor ports, a set of migration model are defined.

Basically, we can categorized various river cells into 5 classes.

a) Entry Cell

The first class is the entry cells of river.

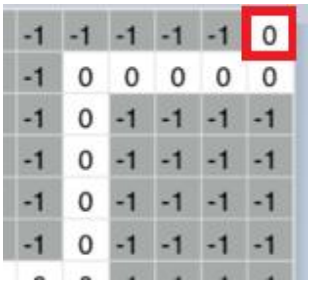

Figure 17. Entry cell of river

As shown above, we can summarize the characteristic of entry cells: the number of neighbor cells with path_v = 0 is 2 and the neighbor cell with path_v = 0 is (0,-1) or (1,0).

Consequently, a migration rule for entry cell can be defined as follow:

```
%entry
rule : { ~migrator_p := $migrator_v; }
{ $display_v := 1;
$number_v := $number_v -
min( ( 1000 - 900 * power( abs( 5000 - time ), 2 ) /
power(5000, 2) ), $number_v );
$migrator_v := min( ( 1000 - 900 * power( abs( 5000 - time ), 2 )
/ power(5000, 2) ), $number_v ) * $probability_v; }
10 { $display_v = 0 and $path_v = 0 and statecount(0,~path_p) = 2
and ( (0,-1)~path_p = 0 or (1,0)~path_p = 0 ) and $number_v > 0 }
```

Figure 18. Migration rule for entry cell

As shown above, the entry cell is in charge of generating input fish to stimulate the migration. A improvement is made on the generation of migration fish where input fish is generated into the river model in a varying frequency. The following equation demonstrates the generation of input fish:

$$Y = 900 * (5000 - X)^2 / 5000^2$$

Y is the number of fish generated while X is the elapsed time units. The graph of output is as below:

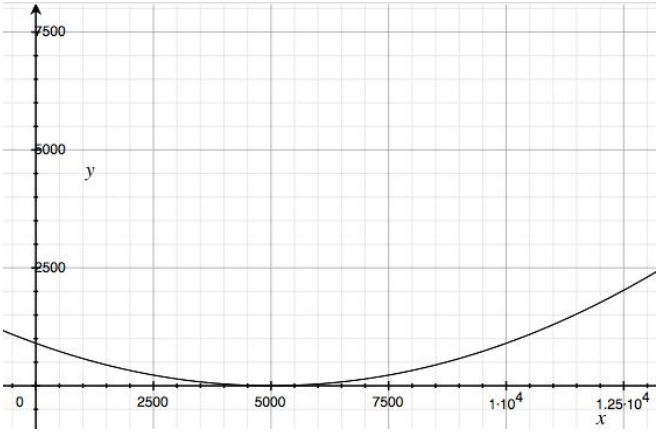

Figure 19. Output graph of function generation fish

b) End Cell

The second class is the end cells.

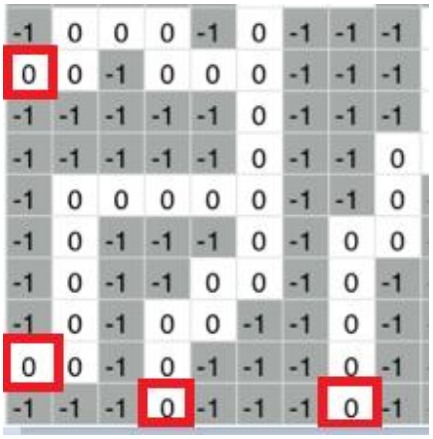

Figure 20. End cells of river

As shown above, we can summarize the characteristic of end cells: the number of neighbor cells with path_v = 0 is 2 and the neighbor cell with path_v = 0 is (0,1) or (-1,0).

Consequently, a migration rule for entry cell can be defined as follow:

```
%end
rule : { ~migrator_p := $migrator_v; }
{ $display_v := 1; $number_v := $number_v + (-1,0)~migrator_p;
$migrator_v := $number_v * $portion_v * $probability_v; $number_v := $number_v; }
10
{ $display_v = 0 and $path_v = 0 and statecount(0,~path_p) = 2 and (-1,0)~path_p = 0 }

rule : { ~migrator_p := $migrator_v; }
{ $display_v := 1; $number_v := $number_v + (0,1)~migrator_p;
$migrator_v := $number_v * $portion_v * $probability_v; $number_v := $number_v; }
10
{ $display_v = 0 and $path_v = 0 and statecount(0,~path_p) = 2 and (0,1)~path_p = 0 }
```

Figure 21. Migration rules for end cell

c) Cell of Main Stream

The third class is cells of main stream.

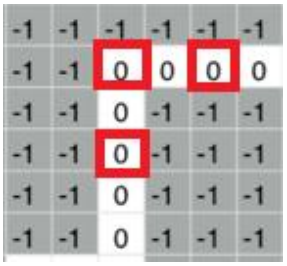

Figure 22. Cells of main stream

As shown above, we can summarize the characteristic of end cells: the number of neighbor cells with path_v = 0 is 3 and there are several combinations of neighbor cells with path_v = 0.

Consequently, a migration rule for entry cell can be defined as follow:

```
rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 3; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and (-1,0)~path_p = -1 and (0,1)~path_p = -1 }

rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 1; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and ( (1,0)~path_p = -1 or (1,0)~path_p = ? )
and ( (0,1)~path_p = -1 or (0,1)~path_p = ? ) }

rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 2; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and (1,0)~path_p = -1 and (0,-1)~path_p = -1 }

rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 2; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and (-1,0)~path_p = -1 and (0,-1)~path_p = -1 }

rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 2; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and (-1,0)~path_p = -1 and (1,0)~path_p = -1 }

%undefined
rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 1; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and (0,-1)~path_p = -1 and (0,1)~path_p = -1
and ( (-1,0)~from_p = 1 or (-1,0)~from_p = 2 ) }

rule : { ~from_p := $from_v; } { $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 3; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0 and statecount(0,~path_p) = 3
and (0,-1)~path_p = -1 and (0,1)~path_p = -1
and ( (1,0)~from_p = 3 or (1,0)~from_p = 2 ) }
```

Figure 23. Migration rules for cells of main stream

d) Crossing Cell With 3 Branches

The fourth class is crossing cells with 3 branches.

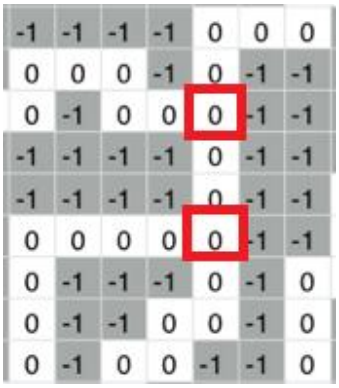

Figure 23. Crossing cells with 3 branches

As shown above, we can summarize the characteristic of end cells: the number of neighbor cells with path_v = 0 is 4 and there are several combinations of neighbor cells with path_v = 0.

Consequently, a migration rule for entry cell can be defined as follow:

```
rule : { ~from_p := $from_v; }
{ $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 2; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0
and statecount(0,~path_p) = 4
and (-1,0)~path_p = -1 }

rule : { ~from_p := $from_v; }
{ $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 1; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0
and statecount(0,~path_p) = 4
and (0,1)~path_p = -1 }

rule : { ~from_p := $from_v; }
{ $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 2; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0
and statecount(0,~path_p) = 4
and (1,0)~path_p = -1 }
```

Figure 24. Migration rules for crossing cells with 3 branches

e) Crossing Cell With 4 Branches

The fifth class is crossing cells with 4 branches.

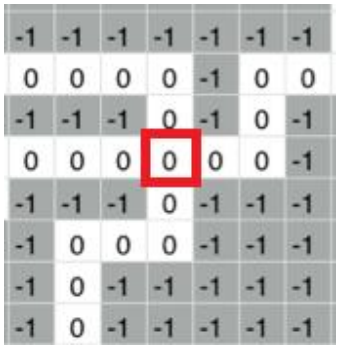

Figure 25. Crossing cells with 4 branches

As shown above, we can summarize the characteristic of end cells: the number of neighbor cells with path_v = 0 is 5.

Consequently, a migration rule for entry cell can be defined as follow:

```
rule : { ~from_p := $from_v; }
{ $branch_v := ( statecount(0, ~path_p) - 2 );
$from_v := 2; $initialize_v := 1; }
10
{ $initialize_v = 0 and $path_v = 0
and statecount(0,~path_p) = 5 }
```

Figure 26. Migration rules for crossing cells with 4 branches

Based on 5 classes of cell summarized above, different set of migration rules are defined to simulate corresponding behaviors. They will be presented respectively in the next section.

VI. SIMULATION

With the model defined above, related simulation are performed under different conditions.

Totally, I build 6 different migration models to demonstrate fish migration under various conditions. They will be presented respectively in the following.

A. Migration_2

This is the simplest model which can be used as the base model.

In this model, probability_v is set to a constant(0.9) for each cell. And portion_v is replaced by a constant(1000), which means the maximum of fish in this cell going to migrate to the next cell remain 1000.

The simulation results are shown below:

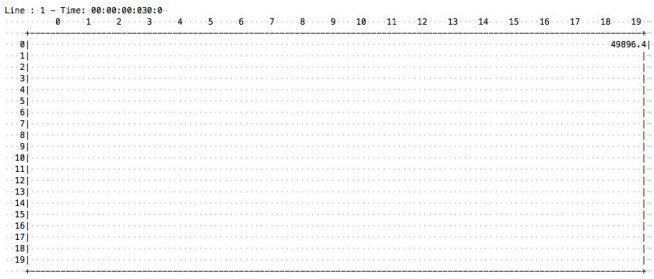

Figure 27. Start of Migration_2 simulation

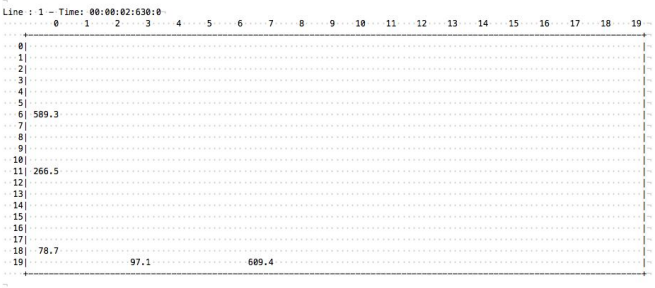

Figure 28. End of Migration_2 simulation

Some screenshots of dynamic migration procedure are shown below:

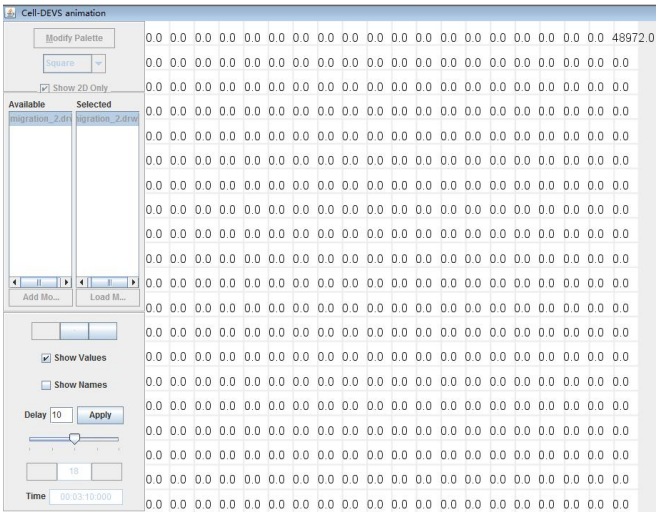

Figure 29. Start screenshot of dynamic Migration_2 procedure

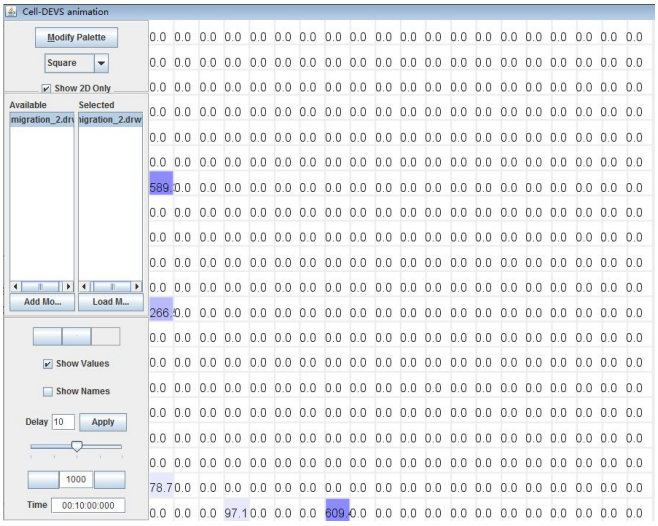

Figure 30. End screenshot of dynamic Migration_2 procedure

B. *Migration_1*

Compared to Migration_2, the portion_v is set to a constant(0.8) in this model, which means only portion of fish inside cell will migrate to the next cell each time.

The simulation results are shown below:

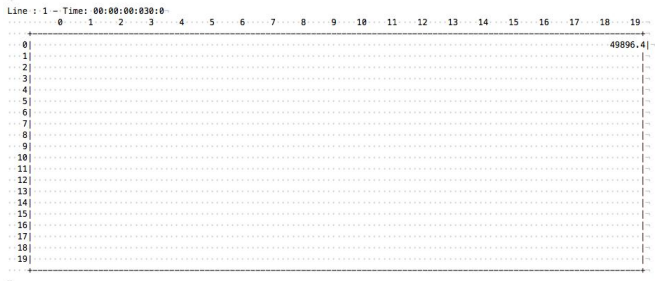

Figure 27. Start of Migration_1 simulation

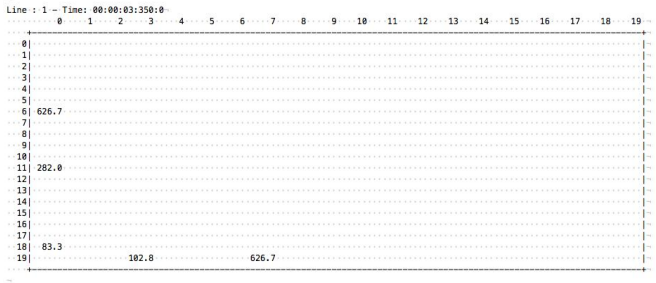

Figure 28. End of Migration_1 simulation

Some screenshots of dynamic migration procedure are shown below:

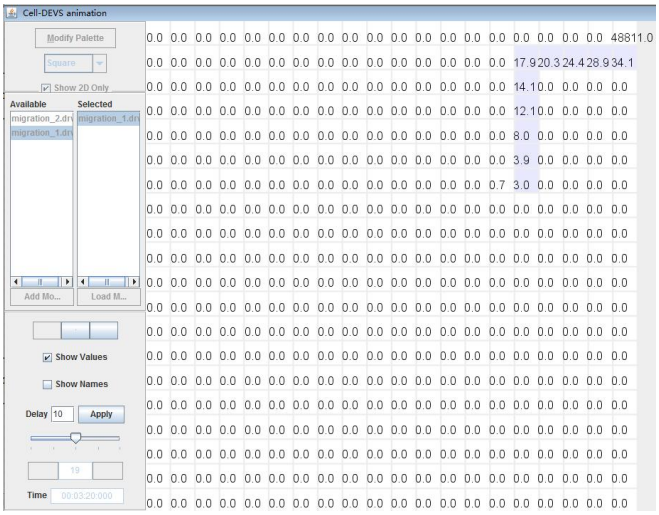

Figure 29. Start screenshot of dynamic Migration_2 procedure

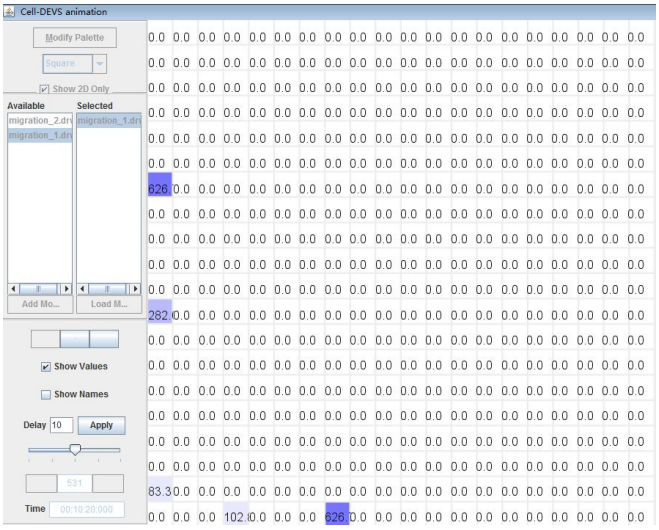

Figure 30. End screenshot of dynamic Migration_1 procedure

C. *Migration_3*

Compared to Migration_1, the portion_v is replaced by a constant(1000), which means the maximum of fish in this cell going to migrate to the next cell remain 1000 each time. Another important change is that probability_v is set to a

function of distance between the current cell and the entry cell. The function is demonstrated as below:

$$y=0.5+0.5\cdot\sqrt{2}\frac{x}{(1+\sqrt{2}x)}$$

Figure 31. Function of probability_v in Migratoin_3

As shown, y is the probability_v and x is the value of cellPos(0).

The simulation results are shown below:

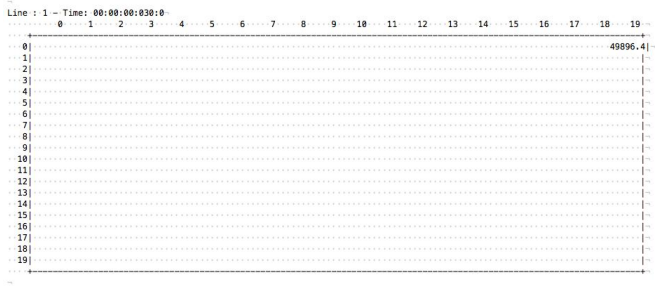

Figure 32. Start of Migration_3 simulation

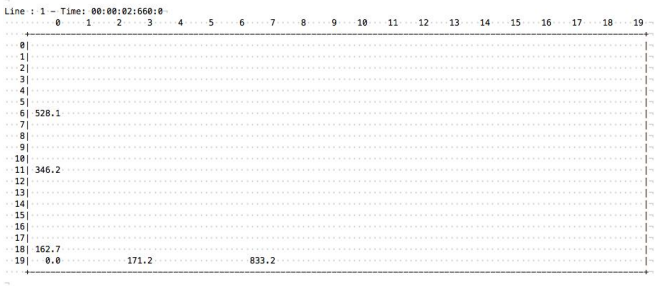

Figure 33. End of Migration_3 simulation

Some screenshots of dynamic migration procedure are shown below:

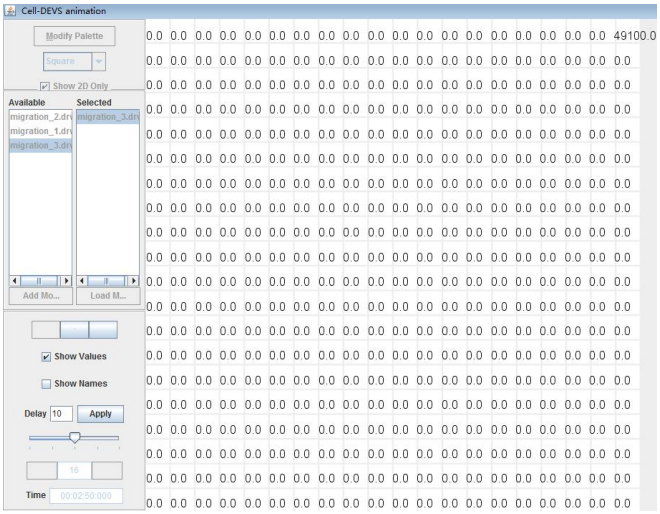

Figure 34. Start screenshot of dynamic Migration_3 procedure

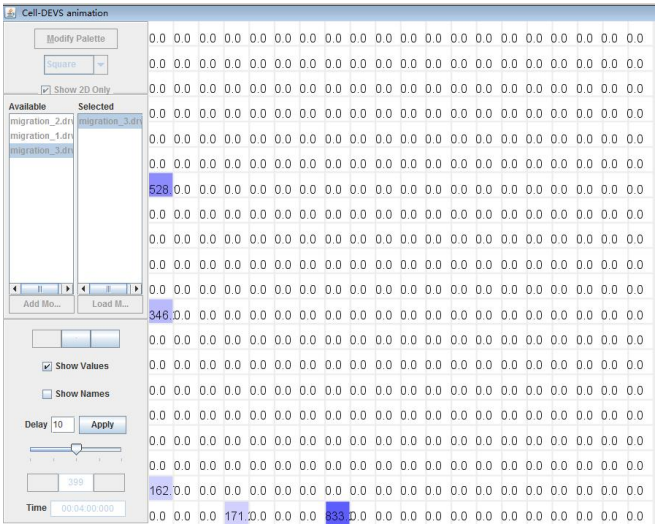

Figure 35. End screenshot of dynamic Migration_3 procedure

D. Migration_4

Compared to Migration_3, in this model, the portion_v the portion_v is set to a constant(0.8) in this model, which means only portion of fish inside cell will migrate to the next cell each time. And probability_v is also set to a function of distance between the current cell and the entry cell, same as the function in Migration_3.

The simulation results are shown below:

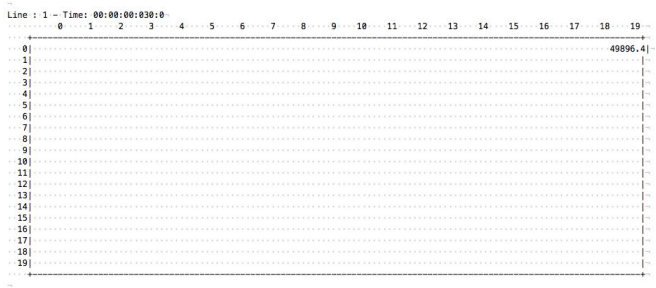

Figure 36. Start of Migration_4 simulation

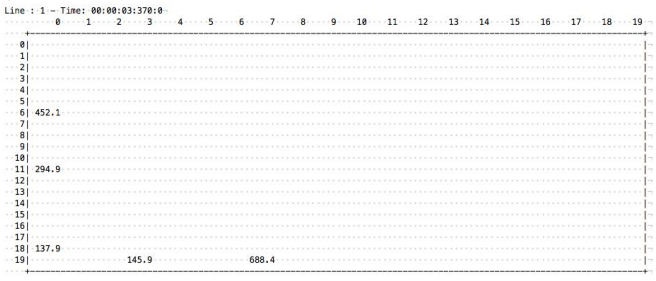

Figure 37. End of Migration_4 simulation

Some screenshots of dynamic migration procedure are shown below:

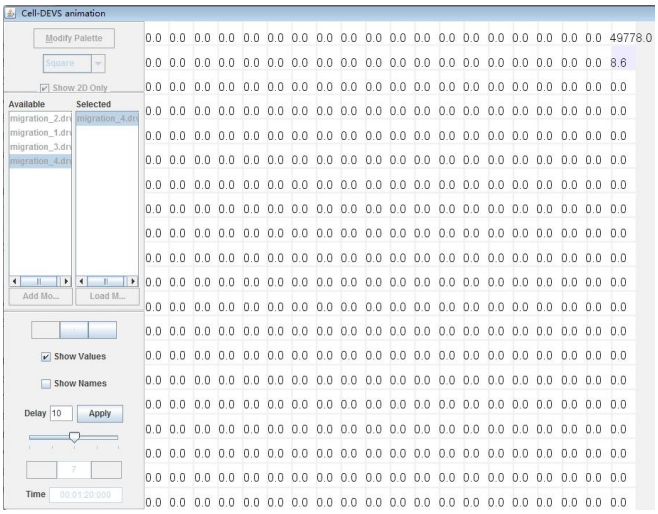

Figure 38. Start screenshot of dynamic Migration_4 procedure

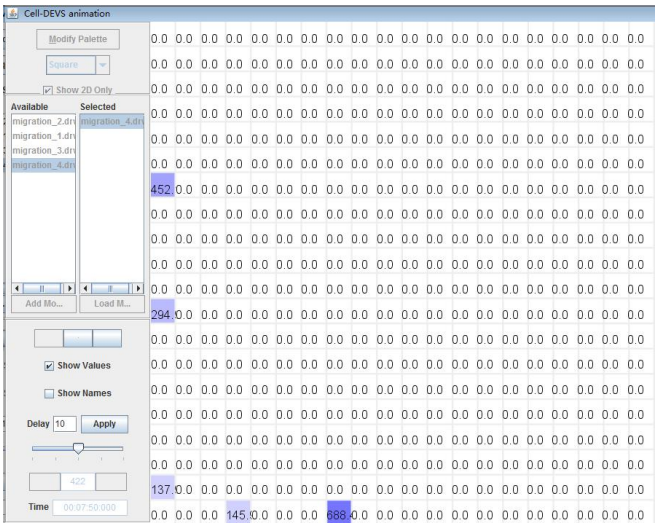

Figure 39. End screenshot of dynamic Migration_4 procedure

E. Migration_5

Compared to Migration_4, in this model, the only change is that the effect of fish number in current cell on the probability_v is added into the function of probability_v.

The new function of probability_v can be demonstrated as follow:

$$Y = Y_0 - X/4000$$

Where Y_0 is the probability_v and X is the fish number inside cell currently.

The simulation results are shown below:

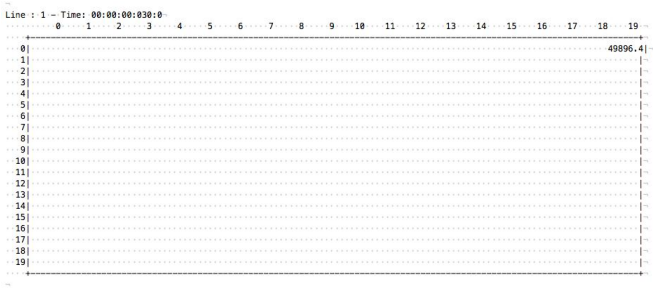

Figure 40. Start of Migration_5 simulation

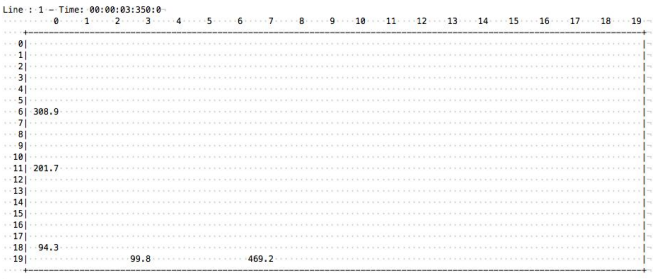

Figure 41. End of Migration_5 simulation

Some screenshots of dynamic migration procedure are shown below:

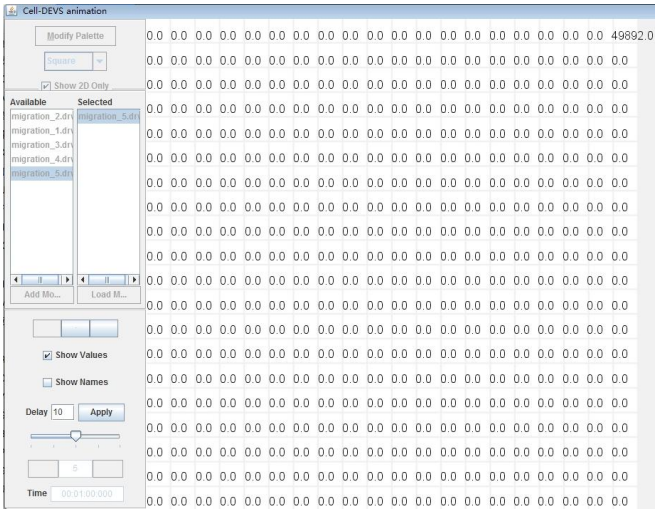

Figure 42. Start screenshot of dynamic Migration_5 procedure

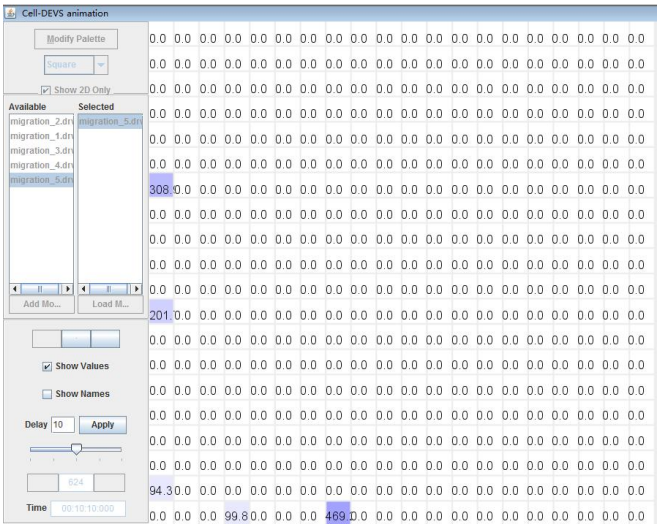

Figure 43. End screenshot of dynamic Migration_5 procedure

F. *Migration_6*

Compared to Migration_5, in this model, the effect of another factor, resource devoted into cell, on the probability_v is added into the function of probability_v.

The new function of probability_v can be demonstrated as follow:

$$Y = Y_0 - X/4000 + Z/4000$$

Where Y_0 is the probability_v, X is the fish number inside cell currently and Z is the resource devoted to improve facility in current cell. The value of Z is initialized in the initialization file(.stvalues) of state variables.

The simulation results are shown below:

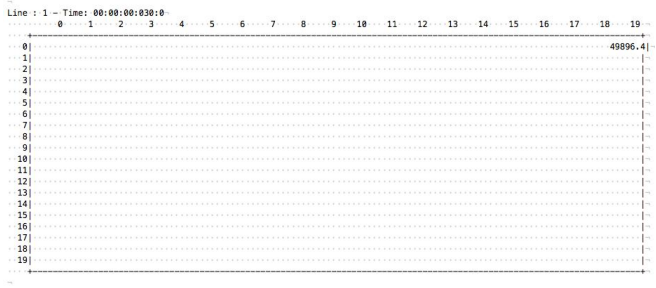

Figure 44. Start of Migration_6 simulation

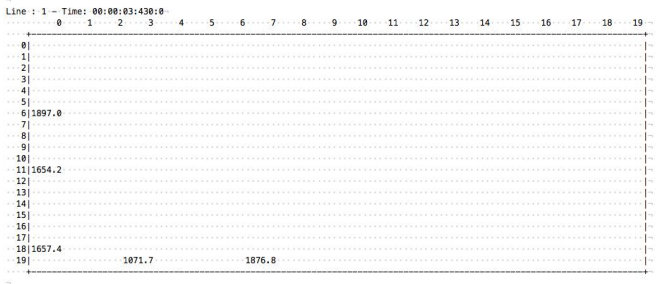

Figure 45. End of Migration_6 simulation

Some screenshots of dynamic migration procedure are shown below:

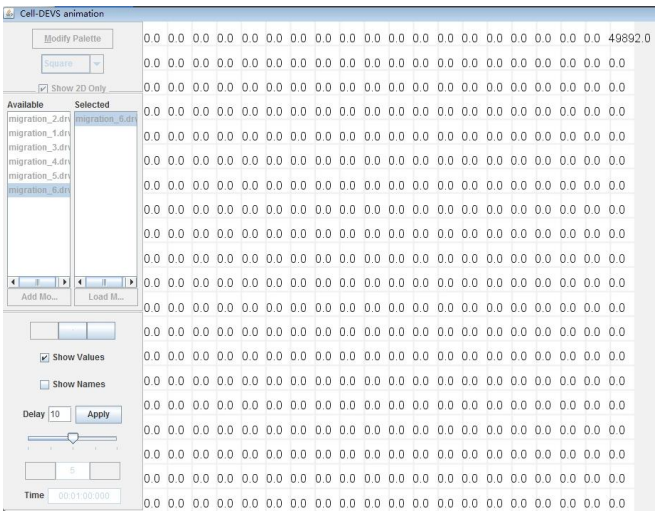

Figure 46. Start screenshot of dynamic Migration_6 procedure

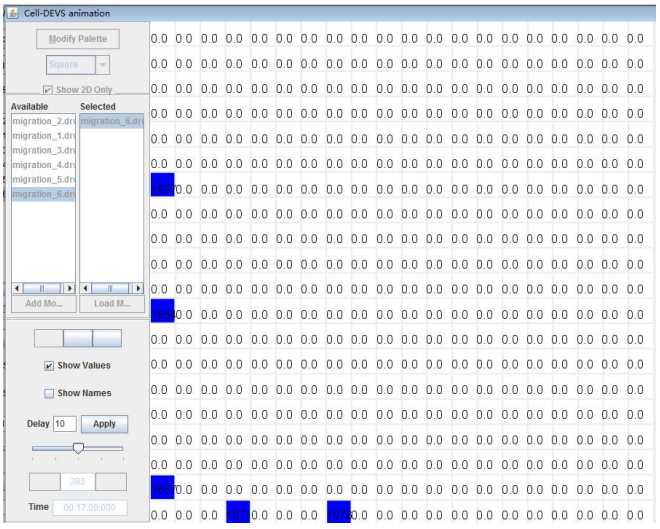

Figure 47. End screenshot of dynamic Migration_6 procedure

G. *Summary and Comparison*

Totally 6 different migration models are built and simulated under different conditions.

The Migration_2 is the simplest one as a base model while the Migration_6 is the most complex one as a final migration model.

In Migration_6, the final function of probability_v can be demonstrated as below:

$$P=0.5+0.5\cdot\sqrt{2}\,\frac{x}{(1+\sqrt{2}x)}-\frac{y}{4000}+\frac{z}{4000}$$

Figure 48. Final function of probability_v

Where P is the final value of probability_v, x is the cellPos(0), y is the fish number inside cell currently and z is the resource devoted into cell.

Comparing simulation results of different models, the following information can be summarized(total number of fish is 50000 at river entry).

Migration model	Fish number at source 1	Fish number at source 2	Fish number at source 3	Fish number at source 4	Fish number at source 5
Migration_2	589.3	266.5	78.7	97.1	609.4
Migration_1	626.7	282.0	83.3	102.8	626.7
Migration_3	528.1	346.2	162.7	171.2	833.2
Migration_4	452.1	294.9	137.9	145.9	688.4
Migration_5	308.9	201.7	94.3	99.8	469.2
Migration_6	1897.0	1654.2	1657.4	1071.7	1876.8

Table 1. Summary of simulation results

VII. CONCLUSION

This objective of this term project is to build a fish migration model with Cell-DEVS to simulate fish migration in river so that different construction strategies could be evaluated and then provides reference when determining the best construction strategy of budget distribution.

With the powerful ability of latest version of CD++, a set of migration model are built, approximating the behaviors of fish migration under various conditions.

From the comparison of simulation results, we can find that with the improvement of facility built over river to facilitate fish migration, the rate of success migration increases remarkably.

Finally, this term project is only a simple step towards a exact and efficient fish migration model. With more modification and upgrade, this model will be able to demonstrate the real fish migration more precisely.

REFERENCES

.

[1] Gabriel A. Wainer, "Modeling and Simulation of Complex Systems with Cell-DEVS", Proceedings of the 2004 Winter Simulation Conference, 2004.

[2] Birgitt Scho nfischl, Michael Kinder, "A Fish Migration Model", 5th International Conference on Cellular Automata for Research and Industry, Switzerland, October 9–11 2002.

[3] Alejandro López, Gabriel Wainer, "Extending CD++ Specification Language for Cell- DEVS Model Definition", Technical Report, April 2003.

[4] Javier Ameghinol, Gabriel Wainer, "Using Cell-DEVS for Modeling Complex Cell Spaces", 13th International Conference on AI, Simulation, Planning in High Autonomy Systems, AIS 2004, Jeju Island, Korea, October 4-6, 2004.

[5] Niloy Ganguly , Biplab K Sikdar , Andreas Deutsch , Geoffrey Canright , P Pal Chaudhuri, "A Survey on Cellular Automata", 2003.