

DEVS-Based Modeling and Simulation for Predator-Prey CSC8840

Anitha Subramanian
Spring 2020



PROBLEM STATEMENT AND OBJECTIVES OF THE PROJECT

Many of the most interesting dynamics in nature have to do with interactions between organisms. One of such interaction is Predator-Prey which are subtle, indirect and difficult to detect. Predator-Prey are the building blocks of ecosystem. Species compete, evolve and disperse simply for the purpose of seeking resources to sustain their struggle for their very existence. This plays an important role in the ecology of populations, determining mortality of prey and birth of new predators. The scope of this project is to identify such kind of dependencies and interactions between the Predator and Prey.

This project presents a DEVS-based Predator-Prey simulation of a Sheep Grass System. The aim of this work is to study how the prey population changes with respect to predator and also to analyze the system behavior by tuning some of predator parameters. The representation in this project can be used to predict prey's growth over time. It also shows that using System Entity Structure to describe a Predator-Prey system is convenient and easy to be modeled in simulation environment such as DEVJAVA.

The objectives of the project are to simulate following scenarios in the well behaving Predator-Prey model,

- Two different predators (with different move time and reproduce time but same life time) that eat the same prey and analyze the growth rate of prey population.
- Restrict the Predator movement directions to 180 degrees and analyze the growth rate of prey population.

GENERAL DESCRIPTION OF THE SYSTEM

The Predator-Prey model has grass as prey and sheep as predator. The initial populations and locations of sheep and grass can be set manually or generated randomly. The grass can grow in any random direction if that selected neighboring place is not occupied by anything. When there is no neighboring free place for the grass to reproduce, then the grass will not reproduce.

The sheep is in constantly moving state. The sheep will search for the grass in neighboring places and move to the place with grass to eat. If there is no grass in the neighboring places, the sheep will move to the empty place with the condition that there should not be any other sheep in it. The sheep should move within specified time units. The sheep will eat the grass immediately after it moves to that grass location. If the sheep doesn't eat grass for specified time units, it will die.

Similar to grass, the sheep will reproduce in any one of the neighboring places if it is free. The sheep will reproduce for every specified time units. Both the sheep and grass need to know the neighboring places to move and reproduce respectively.

GOALS OF THE MODELING AND SIMULATION STUDY

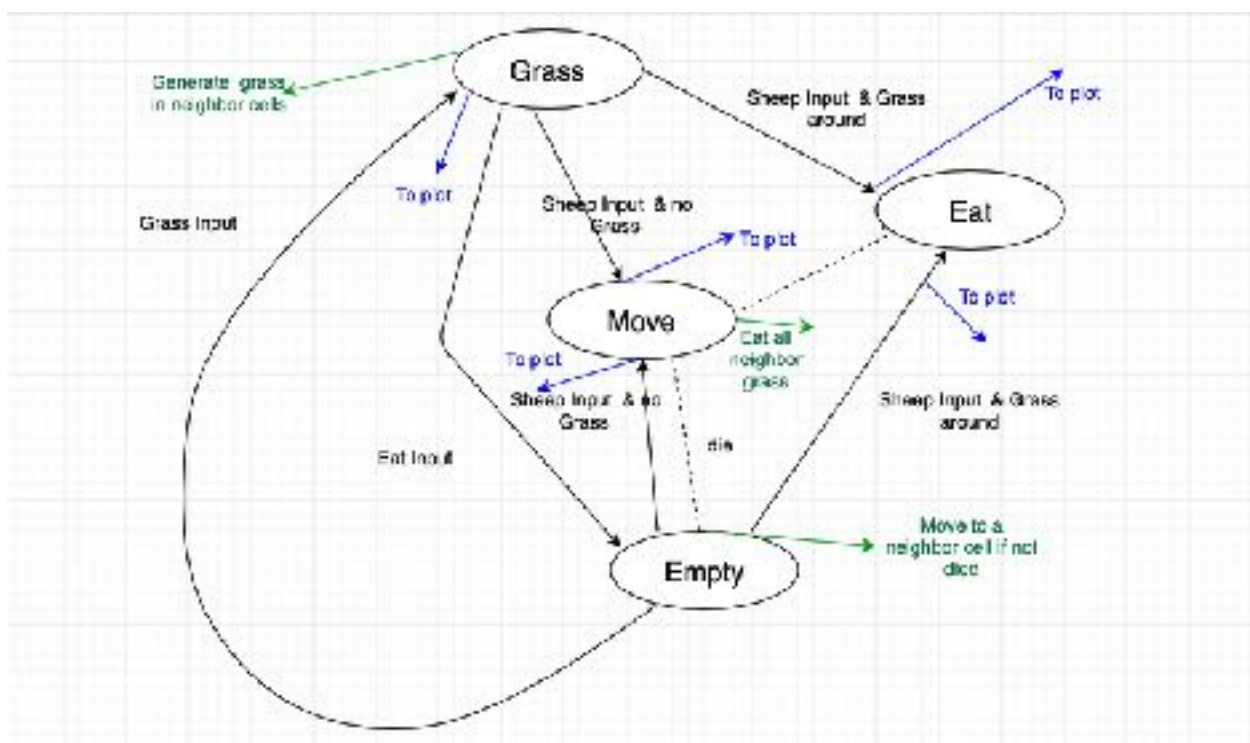
This is a realistic model of a Sheep Grass system. In “real life” the sheep grass model would be characterized by randomly occurring events. The results observed from the simulation of this project could be diverse. Some important aspects that can be measured are,

- Two different predators (with different move time and reproduce time but same life time) that eat the same prey and analyze the growth rate of prey population.
 - Definitely there would be decrease in growth of prey population if there two predators that eats the same prey in the land even if they have different move time and reproduce time.
- Restrict the Predator movement directions to 180 degrees and analyze the growth rate of prey population.
 - If the Predator movement is restricted, there will be increase in growth of prey population.

THE SIMULATION MODEL

MODEL STRUCTURE:

The state diagram for the Sheep grass model is shown below.



The green lines depicts the output messages to the cell space. A sheep cell's attribute (sheep life time and sheep reproduce time) is contained in the message when the sheep moves from one cell to another. Two variables sheeplifetime and sheepreproducetime can be used to keep track of the life time and reproduce time.

EXPLANATION OF OPERATION:

Grass - Modeled behaviors:

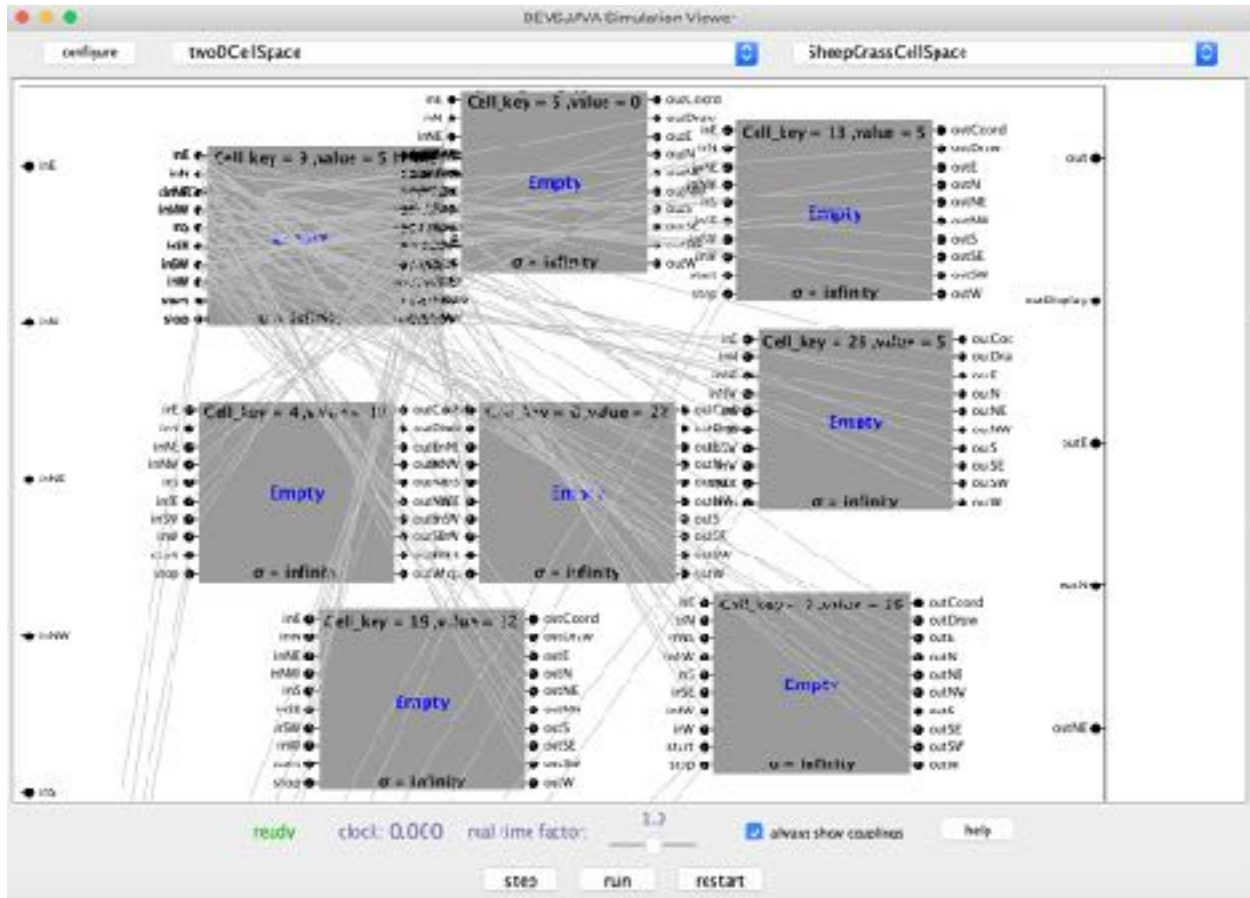
- ◆ Initialization - Hold in grass in random locations of the land.
- ◆ For the phase Grass, hold in grass in the specified x-coordinates and y-coordinates.
- ◆ For the phase Grassreproduce, hold in grass in the specified locations. Get the random neighbor cell out of 8, make the grass reproduce in that cell if it is empty. If it is not empty, don't reproduce.
- ◆ Get the random directions for the grass through the input ports.

Sheep - Modeled behaviors:

- ◆ Initialization - Hold in sheep in random locations of the land.
- ◆ If the phase is Sheep move, check for the grass or empty cell in the neighboring cells and then make the sheep move.
- ◆ For the phase of Sheep or Sheepreproduce, check for the sheeplife time, sheepmove time and sheep reproduce time.
 - ✓ If there is enough life time for the sheep to reproduce (ie., life time is more than moving time of sheep) and enough reproduce time (ie., reproduce time is more than moving time of sheep), switch to sheepmove phase for sheepmovetime. Adjust the sigma time accordingly.
 - ✓ If there is not enough life time for the sheep to move or reproduce, make the sheep die.
 - ✓ If there is more sheep reproduce time than its life time and move time, switch to sheepreproduce phase. Adjust the sigma time accordingly.
 - ✓ Update the sheep move time, sheep life time, sheep reproduce time.
- ◆ Get the random direction for the sheep reproduce or sheep move through input ports.

ILLUSTRATIVE MODEL CODE:

The Sheep Grass Model



DEVs Java code for External transition function of a Sheep Grass model:

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```


DEVS Java code for Internal transition function of a Sheep Grass model:

```

174 public void doInt()
175 {
176
177     if(phaseIs("Grass"))
178         holdIn("Grassreproduce", Grassreproducetime);
179     else if(phaseIs("Grassreproduce"))
180     {
181         System.out.println("grass is producing at " + "x_pos=" + x_pos);
182         grasscount++;
183         grasscount=keepgrasscount(grasscount);
184         System.out.println("Grass count : "+grasscount);
185
186         holdIn("Grassreproduce", Grassreproducetime);
187     }
188     else if(phaseIs("Sheep") || phaseIs("Sheepreproduce"))
189     {
190         if(Sheepmovetime<Sheeplifetime && Sheepmovetime<Sheepreproducetime)
191         {
192             System.out.println("It is coming to sheep state or sheep reproduce state");
193             //System.out.println("Sigma now is="+sigma);
194             holdIn("Sheepmove", Sheepmovetime);
195             Sheepreproducetime=Sheepreproducetime-sigma;
196             Sheeplifetime=Sheeplifetime-sigma;
197         }
198         else if(Sheeplifetime<Sheepmovetime && Sheeplifetime<Sheepreproducetime)
199             holdIn("Sheepdie", Sheeplifetime);
200         else if(Sheepreproducetime<Sheepmovetime && Sheepreproducetime<Sheeplifetime)
201         {
202             holdIn("Sheepreproduce", Sheepreproducetime);
203             Sheepmovetime=Sheepmovetime-sigma;
204             Sheeplifetime=Sheeplifetime-sigma;
205         }
206     }
207     else if(phaseIs("Sheepmove"))
208     {
209         System.out.println("Sheep is moving");
210         SheepGrassCellSpace.getCellCoordinates("lept=");
211         passivateIn("Empty");
212     }
213
214     }
215     else if(phaseIs("Sheepdie"))
216         passivateIn("Empty");
217     else if(phaseIs("Empty"))
218         holdIn("Empty", 0.0);

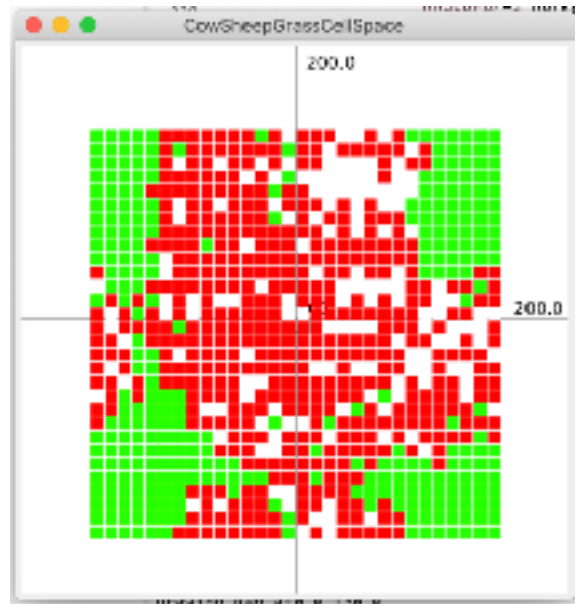
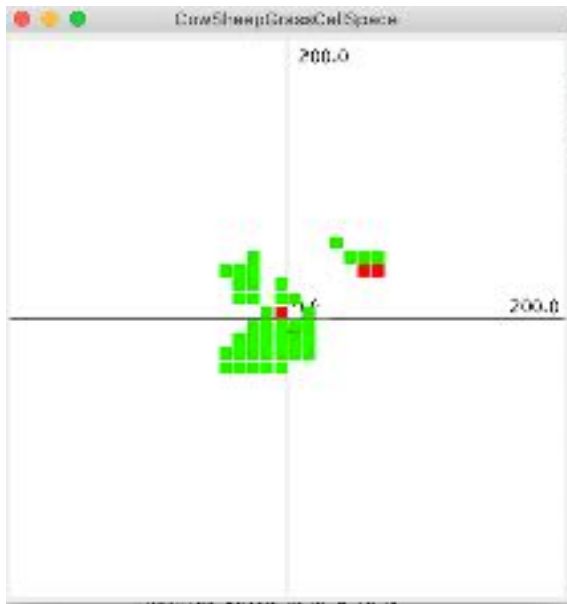
```

sheepMoveT, sheepLifeT, sheepReproduceT, grassReproduceT are the variables used to store the allotted time values for the Sheep move time, Sheep Life time, Sheep Reproduce time and Grass Reproduce time respectively. The dummy variables Sheepmovetime, Sheeplifetime, Sheepreproducetime, Grassreproducetime are initialized with allotted time values and then used for tracking the time values under certain conditions inside the functions (ie. values updated based on sigma value after it reached particular phase).

SIMULATIONS:

The below snapshots shows the simulation of Sheep Grass model. Green represents grass and red represents sheep. Blue represents Cow.

The snapshot on the left shows the initial setup of the Sheep grass model with few grass and 3 sheep in the land. The snapshot on the right shows the stable oscillation of sheep and grass after some time.



EXPERIMENTAL DESIGN:

The following experimental frames can be developed for this Sheep Grass model.

1. Cow Sheep Grass experiment

- This experiment can be made to analyze the growth rate of prey population when there is a single predator that eats the prey vs two predators that eat the same prey. This can be achieved by adding one more predator to the system with different parameters. So I have taken Cow as my other predator.

Initial conditions - Some cells with grass , few sheep and few cows.

Input generators - Phase received from the input ports such as Sheep move, Cow reproduce, etc.,

Simulator - Simulates the movement and reproduction of two predators by eating the same prey in the land. Two predators cannot eat the same prey. They can eat only the prey is available without any predator on it.

Time for Predators - The reproduce time, move time are different for both the predators except the life time. This allows us to identify the affected growth of prey population clearly.

Expected result - The growth of prey population will be less over the period of time because of more number of predators even if they have different parameters.

Code Snippet for Cow Sheep Grass experiment:

```
public void scenario10() //CellSpace with both Sheep and Cow
{
    if(x_pos==10 && x_pos<10 && y_pos==10 && y_pos<10 || x_pos==40 && x_pos<50 && y_pos==10 && y_pos<10 || x_pos==40)
    {
        holdIn("Grass", 0);
        CowSheepGrassCellSpace.reInitPhase(x_coord, y_coord) = "Grass";
    }
    else if(x_pos==20 && y_pos==10 || x_pos==20 && y_pos==10)
    {
        holdIn("Cow", 0);
        CowSheepGrassCellSpace.reInitPhase(x_coord, y_coord) = "Cow";
    }
    else if(x_pos>20 && x_pos<40 && y_pos==10 && y_pos<10 || x_pos==40 && y_pos==10 || x_pos==40 && y_pos==0)
    {
        holdIn("Sheep", 0);
        SheepGrassCellSpace.reInitPhase(x_coord, y_coord) = "Sheep";
    }
    else
        possibleIn("Busy");
}
```

2. Sheep Direction Restriction test

- This experiment also made to analyze the growth rate of prey population by restricting sheep direction to 180 degrees instead of 360 degrees. ie., Restricting sheep move and sheep reproduce actions to only 5 directions instead of 8.

Initial conditions - Some cells with grass and few sheep. Change the functions getdirectionsheepreproduce, getdirectionsheep with directions limiting to 5.

Input Generator - Phase received from the input ports such as Sheep move, Grass reproduce, etc.,

Simulator - Simulates the restricting movement of sheep to particular directions.

Grass direction - This would be all the 8 directions unlike sheep.

Expected result - The growth of prey population will be more as there is a restricted movement of predator in the land. Also there will be less population of predators since the reproduce direction also restricted.

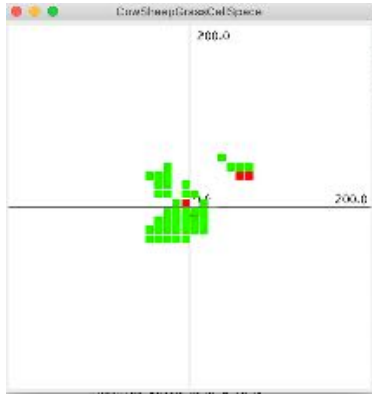
Code Snippet for Sheep Direction Restriction test:

```
1 // Scenario 10: Sheep Direction Restriction test
2 // This scenario tests the effect of restricting sheep movement to 5 directions.
3 // The initial conditions are:
4 // - 10 cells with Grass (x=10, y=10 to x=40, y=10)
5 // - 10 cells with Cow (x=20, y=10)
6 // - 10 cells with Sheep (x=40, y=10 to x=50, y=10)
7 // The input generator provides the following phases:
8 // - "Grass" (x=10, y=10 to x=40, y=10)
9 // - "Cow" (x=20, y=10)
10 // - "Sheep" (x=40, y=10 to x=50, y=10)
11 // The simulator restricts sheep movement to 5 directions:
12 // - Up (y-1)
13 // - Down (y+1)
14 // - Left (x-1)
15 // - Right (x+1)
16 // - Diagonal (x-1, y-1)
17 // The expected result is that the prey population will grow faster than in the unrestricted case.
18 // The predator population will be lower than in the unrestricted case.
19 // The simulation ends when the prey population reaches 100.
20 // The output is the number of prey and predator cells at the end of the simulation.
```

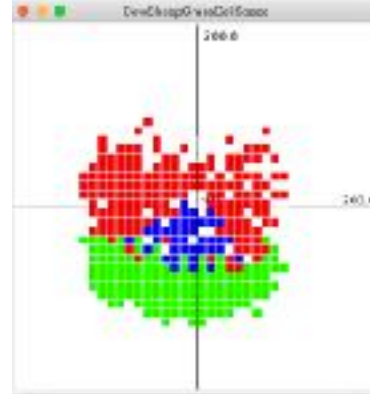
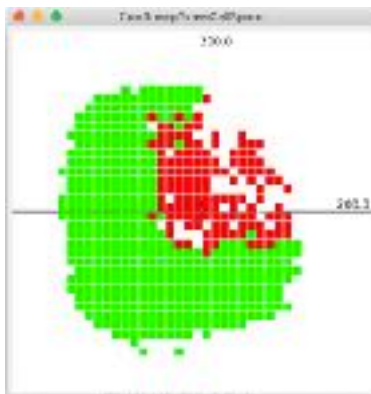

SIMULATION RESULTS & ANALYSIS:

Cow Sheep Grass Experiment:

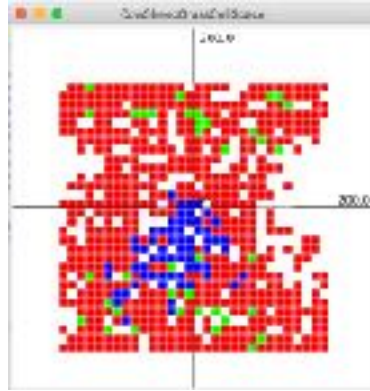
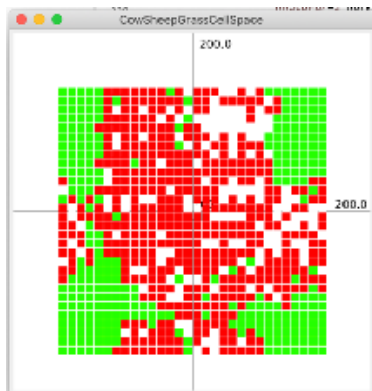
1. Initial state - Sheep Grass System Vs Cow Sheep Grass System



2. After time t - Sheep Grass System Vs Cow Sheep Grass System

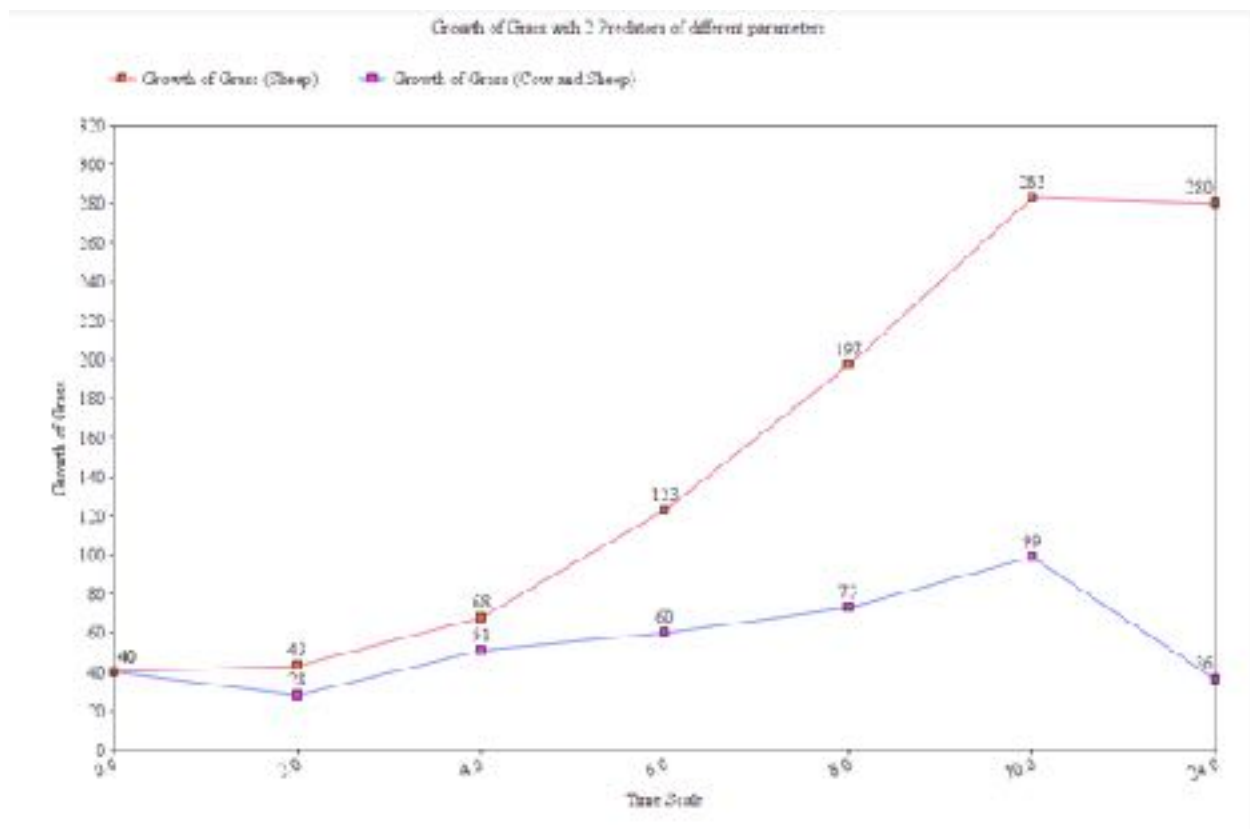


3. At the end - Sheep Grass System Vs Cow Sheep Grass System



Time value (e)	Growth of Grass (sheep)	Growth of Grass(Cow and Sheep)
Initial	40	40
2.0	43	28
4.0	68	51
6.0	123	60
8.0	197	73
10.0	283	99
Towards end	280	36

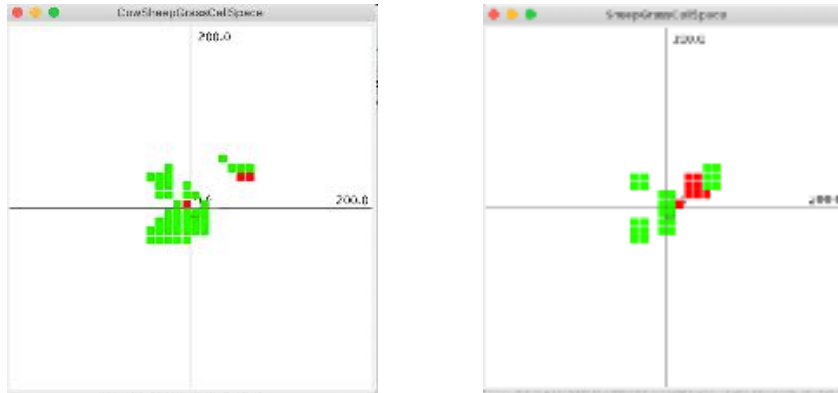
From the above simulation results data, We can see that the growth rate of prey population is affected when there are two predators with different parameters that eats the same prey in the system compared to single predator in the system.



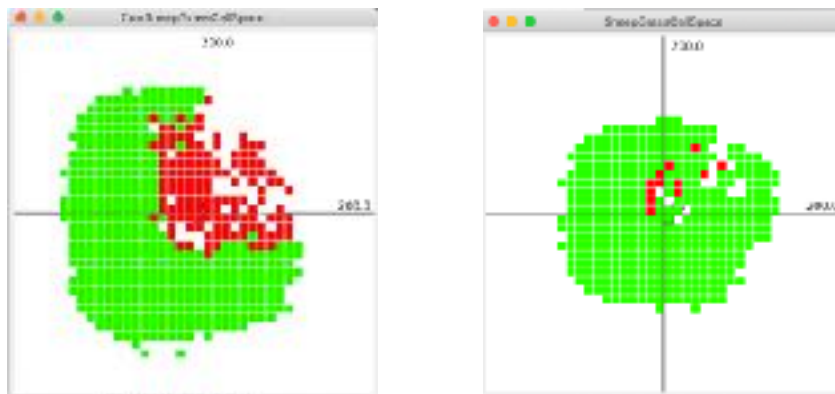
From the above graph, we can see that there is a fall in growth of grass with 2 predators of different characteristics.

Sheep Direction Restriction test:

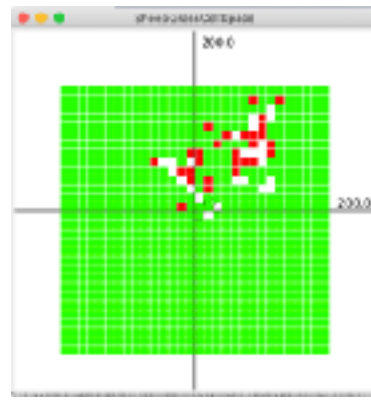
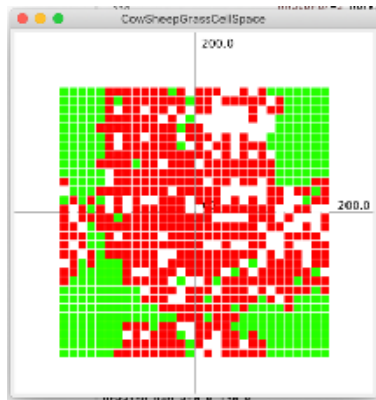
1. Initial State - Sheep with 8 directions Vs Sheep with 5 directions



2. After time t - Sheep with 8 directions Vs Sheep with 5 directions

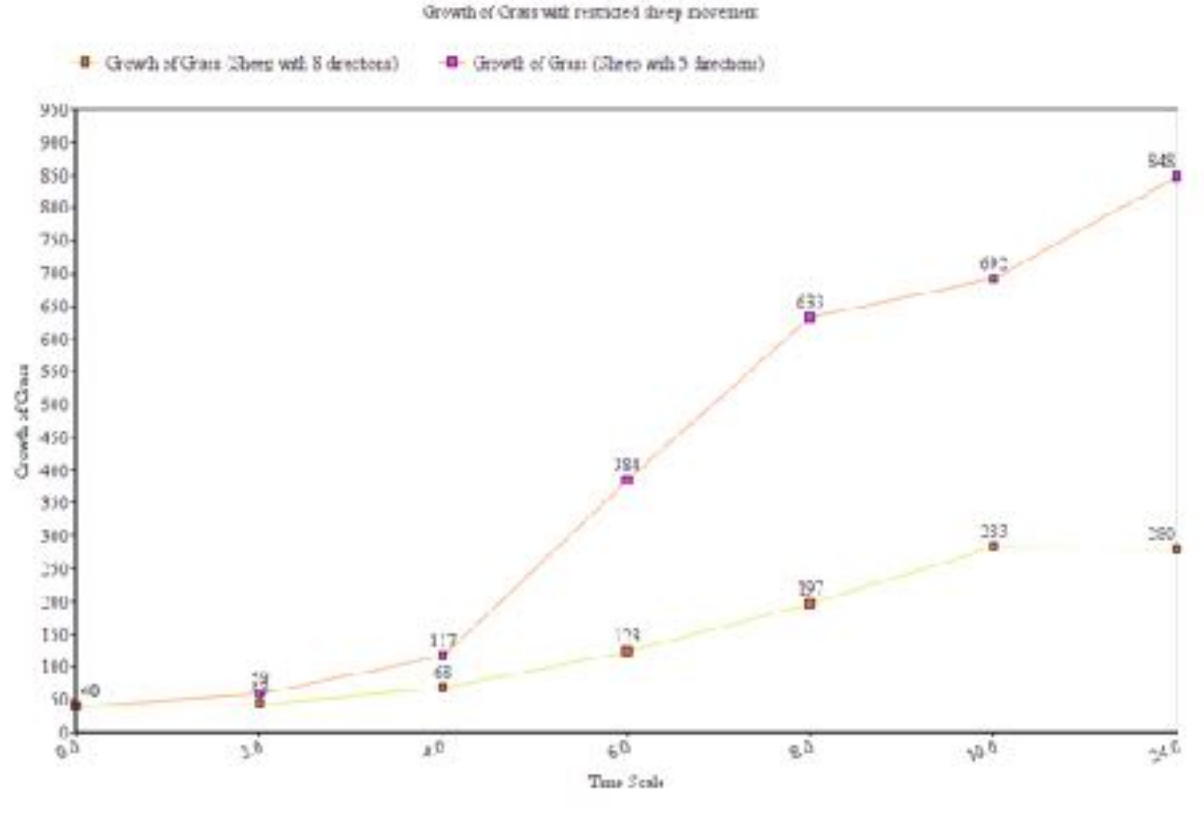


3. At the end - Sheep with 8 directions Vs Sheep with 5 directions



Time Value (e)	Growth of Grass (Sheep with 8 directions)	Growth of Grass (Sheep with 5 directions)
Initial	40	40
2.0	43	59
4.0	68	117
6.0	123	384
8.0	197	633
10.0	283	692
Towards end	280	848

From the above simulations data, we could see that restricting the direction of predator to limited directions results in high growth rate of prey population and also very less predator reproduce.



From the graph, we can see that the growth of prey is more with the restricted movement of predator. If it is not restricted, the growth of prey will be less as seen from the graph.

CONCLUSION:

We could observe from these simulation experiments that the growth of prey population depends on predator and its parameters. When we add one more predator that eats the same prey to the system, there is a less growth rate of Prey population. So the predator population is indirectly proportional to prey population. *Higher the Predator lesser the Prey.* Another simulation experiments shows that restricting the movement of predator results in high growth rate of prey population. *When there is a restriction on Predator, Prey population increases.*

FUTURE WORK:

This model is just implemented with single predator and prey. The future work may include adding different predator to the system like wolf and observe the system under the same above mentioned conditions.

REFERENCES:

1. CSC8840 - Our Class slides and documents from the Professor Dr.Xialion Hu.
2. http://www.scholarpedia.org/article/Predator-prey_model - For basic understanding of the Predator-Prey model.
3. Zeigler, B.P., Praehofer, H., and Kim, T.G. (2000), theory of modeling and simulation. 2ed, Academic Press, New York, NY.