Alex Ziegenhorn

Project Report

Due: 04/26/2017

**Design a rational problem-solving agent for an interesting application. Two algorithms – genetic algorithm (GA) and simulated annealing (SA) – should be used independently to solve the problem. In the project report, include the following:**

**What is your application problem?**

**Initial State:**

The predator spawns in the upper-leftmost corner of the grid, the prey spawn randomly throughout the grid.

**Actions ( ACTION(s) ) and Transition Models ( RESULT(s, a) ):**

The predator has a set of actions that it can carry out every tick:

- Move: Moves one tile adjacent of current position of the predator.

Formal Action: Position(GridTile), {Move(NeighboringGridTile)}

Formal Result: Position (GridTile), Move(NeighboringGridTile) = Position(NeighboringGridTile)

- Drink: Only usable on water tiles; predator drinks to relieve thirst value.

Formal Action: Position(GridTileNeighboringWaterTile), {Drink()}

Formal Result: Position(GridTile), Drink() = (self.thirst = 100.0)

- Attack: If at least one prey exists in an adjacent tile, predator attacks prey, prey disappears, and predator moves to prey’s previous location.

Formal Action: Position(GridTileNeighboringPrey), {Attack()}

Formal Result: Position(GridTileNeighboringPrey), Attack() = (prey.removed , Move(prey.position)

**The prey’s actions:**

- Move: Moves one tile adjacent of current position of the predator.

Formal Action: Position(GridTile), {Move(NeighboringGridTile)}

Formal Result: Position (GridTile), Move(NeighboringGridTile) = Position(NeighboringGridTile)

**Goal Test:**

Has the predator kept his hunger and thirst bars above 0% for the duration of the program?

**Path Cost:**

The number of ticks spent for the action taken to reach the next node/state.

**Describe the representation of your problem that is used to solve the problem using GA.**

I use the genetic algorithm when the predator must decide which path to take to get to a water tile in the grid. Using the genetic algorithm in this way returns an entire path, which the predator can then use to travel efficiently to the closest water source.

**Describe the representation of your problem that is used to solve the problem using SA.**

I use simulated annealing when the predator must decide which path to take to get to a prey object in the grid. Since the prey are constantly changing location, I have the simulated annealing algorithm simply return the best possible move *towards* the prey, so that the path is updated after every move by the predator.

**Mathematically define the fitness function for your problem for GA.**

In code, it is represented as:

-**SimulatedAnnealing**.*distanceFormula*(finalLoc, **this**.goal)-(**this**.maxDistance/100);

Now to break it down:

* First, we take the distance between the final location of the path and the goal location as a negative value (distance)
* Then, we take the maximum number of moves allowed for a population and divide that by 100, so that we can have a more precise fitness value for use in comparisons (maxDistance/100)

Mathematical Representation:

-distance – (maxDistance/100)

**Mathematically define the** **objective function for your problem for SA.**

In code, it is represented as:

**Math**.*exp*( (currentEnergy - newEnergy) / temperature );

Now to break it down:

* First, the current energy of the best solution (distance)
* Then, we take the maximum number of moves allowed for a population and divide that by 100, so that we can have a more precise fitness value for use in comparisons (maxDistance/100)

Mathematical Representation:

-distance – (maxDistance/100)

**Compare the two algorithms using the accuracy of your solution.**

**Compare the two algorithms using the time (in seconds) required to reach convergence. Show convergence plot for each algorithm. Calculate time taken as discussed in class today (4/21/2017).**