Adaptive Resource Gathering & Trading

Konstantinos Sideris, Shankar Manjunatha

March 22, 2012

Abstract

Individual animats trading independently can lead to a collective market behavior. Also, animats can be imbued with gathering capabilities, enabling it to survive in a resource-filled environment. We separately develop animats that are capable of trading and resource-gathering, and run experiments to determine the conditions for survivability of the animats. We then imbue all animats with trading and gathering skills, and run experiments to determine the survivability of animats under controlled and random environments. These experiments confirm our understanding market behavior, as well as providing new insight.

Introduction

• We modeled a 2-resource (food, water) environment and market with multiple animats.

• The animats are equipped with sensors (for food, water, exchange price), and effectors (to move, gather, trade).

• The goal of each animat is to gather and trade resources in a way to maximize its total resources.

• We developed the market and the gathering environment independently, and ran experiments to test the behavior of animats in these environments.

• We then combined the two environments, and ran experiments to see the behavior of the animats, both in controlled and random environments.

• The results obtained confirmed our hypothesis, and some results provided additional insight into the outcome of markets in environments with limited resources.

Hypotheses

1. In the Trading Environment.: The animats exhibit adaptive behavior via bidding which mimics that of actual humans.
2. In the Gathering Environment: The animats are expected to gather the resource that they have less of, but if there is another resource which is closer/on-the-way, that should be taken into account.

3. In a mixed environment: They combine the resource gathering and trading capabilities to gather the more abundant (local) resource and trade it for the other resource, and thus maximize their overall resources.

**Goals, Issues, Problems**

The overall goal for each animat is to maximize its resources, and thus its life expectancy.

This is split into the following sub-goals:

- **Trading Environment:**
  - To exchange its more abundant resource for its less abundant resource.
  - Obtain the best possible price for the exchange.
  - Manage the above, to maximize life expectancy: If it holds out for a very good price, it may not complete the trade, and die out due to unequal resources. If it trades for a bad price, it sacrifices too much of the abundant resource, and shortens its life expectancy.

- **Gathering Environment:**
  - To gather the less abundant resource at each stage
  - To not make wasteful trips: If the animat requires food, but sees water nearby, it should gather water first
  - The conflicting goals to be managed so that in gathering water, the animat doesn’t die out

- **Mixed Environment:**
  - In certain controlled experiments, the animats must be able to decide between gathering and trading as a means to increasing the levels of its less abundant resource.
  - The animats should be more “prosperous” in mixed environments compared to gathering-only environment

**Previous Work**

We did not use any pre-existing code/work from any source, but some of our ideas were influenced by these papers:
For trading:
1. Beltratti and Margarita (1992) : Evolution of trading strategies : The authors describe animats that attempt to predict a stock price.
   - Animats build an expectation of the stock price based on inputs such as previous day stock price, the number of shares it owns, and the money it has.
   - The traded price depend on these expectations
   - We used some of their ideas about separating the market from the animat, and completing one trade for each animat in a single timestep.

2. Cliff and Bruten (1999): Animat Market-Trading Interactions as collective social behavior
   - We used their idea of having Bids and Offers

3. E. van der Vaart, B. de Boer, A. Hankel, B. Verheij : Agents Adopting Agriculture-Modeling the Agricultural Transition
   - We used their idea of a free-distribution model, which says that an environment can be divided into smaller, discrete habitats
   - Their experiments of Gathering only, Hunting & Gathering, as well as Hunting, Gathering & Farming, provided us with an idea of how to design and perform our experiments.

Environment & Physics

- Our world consists of a terrain populated with two types of resources, water and food. It consists of a grid of habitats which contain a certain number of water and food (figure 1).

- Our agents are an abstract form of intelligent animals. Each agent represents a single animal and shall be called animat for the rest of the document. An animat needs both types of resources to survive so its goal in terms of survival is to maximize both with an emphasis in minimizing their difference.

- In this environment the time unit is defined as a single day. During each time unit an animat consumes one unit of food and one unit of water to sustain itself. Animats can collect only one of the two types of resources in a single time unit as well as trade one resource type for the other.

- There is no limit to the carrying ability of the animats. Animats can move from habitat to habitat (only in straight lines, not diagonally) but there is a resource cost associated with this action.

- Since there are only two resources, there is no need for a currency. Instead, a price is expressed in terms of units of food per units of water.
Methodology

We developed our solution in three phases. First, we implemented trading animats, then we designed and tested gathering animats and finally we combined them into a single animat.

1 Trading

Brain of a trading animat:

The brain of a trading animat is modeled with sensors, effectors and nodes with input/outputs. Price setting is performed as follows (figure 2): The animat has one sensor for each stored resource. With these sensors it calculates the minimum resource which is the type of resource it wants to acquire. The overall goal for the animat is to minimize the difference between its resources by trading the excess resource for the other one.

At the beginning of each trading round, an animat decides on the type of resource it wants to sell/buy as well as the desired quantity and price. After all the animats have provided their bids, trading occurs. The price that the animat sets is initially 1 unit of the excess resource for the difference in resources. In that sense, if the trade was achieved it would immediately equalize its resources. As it can be seen in the equations above, we introduced an additional factor in the price. This factor represents the need for the minimum resource. If the animat has enough of both it can afford to wait. If, however, one of the resources is almost depleted then it has to lower the price to increase the chances of a successful trade. The quantity is also adjusted to fit this adjusted price. That is, if the new price is met, then the animat will get the quantity it initially required.

- \( \text{Price} = \frac{1.0}{(\|\text{water} - \text{food}\| \times \min(\text{water}, \text{food})/100.0)} \), if selling food
- \( \text{Price} = (\|\text{water} - \text{food}\| \times \min(\text{water}, \text{food})/100.0) \), if selling water
- \( \text{Quantity} = \frac{||\text{water} - \text{food}|| \times \text{Price}}{\text{Price} + 1.0} \)

Initially, we had designed the trading to be performed by individual animats. Every animat would take a chance to examine the offers and accept or reject them. While we achieved trading in this scenario, the results showed that not all animats got a fair deal out of the trading since priority played a big role. This does not reflect how markets work. In a large enough market scenario, every agent will try to maximize its profit from the transaction. To model this in our environment, it would require multiple rounds of trading with every animat varying its price and quantity in order to achieve the maximum profit. Instead, we chose to model this collective behavior by having an outside algorithm simulate this process. That is, the market defines who is going to trade with whom and at what price mimicking an actual market.
while (trades are possible) :

\[
x = \text{FindMinimumPriceAmongFoodSellers}
\]

\[
y = \text{FindMaximumPriceAmongWaterSellers} // \text{For Localized Trading, this part is restricted to traders within range}
\]

\[
\text{trading price} = \sqrt{\text{price}_x \times \text{price}_y} \text{ (geometric mean)}
\]

\[
\text{trading quantity} = \min(\text{quantity}_x, \text{quantity}_y)
\]

| Table 1: Trading Algorithm |

## 2 Gathering

Brain of Gathering Animats:

The type of controller is similar to the trading animats. The animat senses resources by an abstract smell sensor. The smell sensor identifies the most intense smell which in turn is determined by the distance and the size of the resource:

\[
\text{smell} = \frac{\text{Quantity}}{\text{Distance} \times \text{MOVING}_\text{COST}}
\]

The overall brain of a gathering animat can be seen in figure 3. The output of the intermediate GoToWater, GoToFood nodes are sigmoid functions of their inputs as we determined experimentally that sigmoid function have bigger discriminatory power. The Motor effector abstracts the gather/move operations. It either moves the animat towards the resource, or, if the animat is on top of it, it collects it. The motor effector is activated based on max(GoToFoodOutput, GoToWaterOutput).

\[
\text{GoToFoodOutput} = \frac{1}{1 + \exp(-\frac{\text{FoodSmell}}{\text{StoredFood}^3})}
\]

\[
\text{GoToWaterOutput} = \frac{1}{1 + \exp(-\frac{\text{WaterSmell}}{\text{ StoredWater}^3})}
\]

We also determined experimentally that the cube of the stored resource is an adequate function to characterize the need for that resource. The square was not enough (the animat was drawn to the smell mostly) and the fourth power was too intense (the animat was only concerned with its minimum resource, ignoring smell).

## 3 Trading & Gathering

Combining the two types of animats above in a single animat was fairly straightforward. The combined brain of trading & gathering animats is the same as the individual ones minus the redundant sensors. An action we had to take was to introduce locality into trading to be able to simulate localized markets.
We made many different considerations for how to combine trading and gathering. We tried various types of interaction, with sensors that introduced trading prices into the gathering decisions and vice versa. The results were not satisfactory as the animat would not make the best choice for its survival. On the other hand, as trading and gathering have separate goals (trading tries to minimize difference, gathering collects the most ‘smelly’ resource) we determined that it is best to keep the actions independent. In a single day, an animat tries to collect a resource and participates in a trading round.

Instrumentation

Common to all experiments:
- List of animats (identified by a unique ID) and state (alive/dead)
- Food and water resources of each animat
- Survivability = average (min resource for every animat)
- Prosperity = average (sum of resources for every animat)

Trading Environment:
- IDs of the animats which traded
- Trade price for each trade

Gathering Environment:
- Distribution of resources in the environment
- Location of each animat: (x,y) co-ordinates

A mixed environment saves the information present in both the trading and gathering environment.

Experiments

To verify our hypotheses, we ran three separate experiments. We let the experiments run until all animats die (this happens when one of their resources drops below 0). The simulation parameters in all experiments were:
- COLLECT_RATE = 5
- FOOD_CONSUMPTION_RATE = 1
- WATER_CONSUMPTION_RATE = 1
- MOVING_FOOD_COST = 5
- MOVING_WATER_COST = 5
1. Large Environment with many agents (with and without trading):

This experiment was designed to showcase that in a random, lush environment our animals will benefit collectively from trading. That is, the survivability index and prosperity index in our environment will be greater if we introduce trading. In this case the habitats where initialized with random food/water between 0,100 (the same for all runs though). Figure 4 demonstrates the prosperity and survivability indices as they evolve during the simulation.

From the graphs, it is obvious that our animals benefit collectively from trading. During most of the simulation time, the animals are on average more prosperous and more likely to survive also. One interesting observation is that by not trading, a small number of animals (1-2) survive slightly longer. It is interesting because it shows that not interacting with the rest of the group can sometimes be beneficial on an individual level.

2. Isolated Islands (with and without trading):

Our second experiment aimed to demonstrate what happens when animals are located in remote environments that only contain one of the two necessary resources. We initialized the environment with habitats on two corners (top left and bottom right) with only water or food, but plenty of that resource. We introduced four agents, two at each corner. The results can be seen in figure 5.

In this experiment, the animals survived much longer when trading was introduced. It is fairly obvious that without access to one of the resources and no trading, they will die out really quickly.

3. Isolated Islands (Global and Local trading):

Our third experiment aimed to analyze the differences between local and global trading. We initialized the environment with habitats on all corners and agents placed on those environments. The results can be seen in figure 6.

Local trading here implies that the animals in a particular resource-rich block can only trade between themselves. And global trading implies that each animal can trade with all other animals in the environment. This was one case which may provide new insight into economics, particularly in the case of a controlled, limited-resource environment. The difference in the survivability and prosperity indicate inequality between the animals. We found that there is more inequality when there is global trading.

Our best conjecture was that having more actors in the global trade introduces more inequality in the market. We did not have the time to search for any economic papers about this, but this would be an interesting direction for future work.
Current status of work

We completed all that we set out to do in the initial proposal. We even provide an interesting direction for future work in the Experiments Section.

Implementation

Contribution

Konstantinos independently contributed the resource gathering part. Shankar independently contributed the trading part. Then, both of us worked together to combine the two to create the mixed environment. This was necessary, as there was a need for improvement in both the parts for the combined environment to work.

Language/tools/packages

For our implementation we decided to work exclusively with Python (version 2.7) under Ubuntu Linux (version 11.04). This choice was made consciously since Python is an interactive scripting language that allows for fast prototyping and at the same time supports a variety of useful libraries and is sufficiently fast for our purposes. Our graphical interface was built from scratch using the game engine PyGame (version 1.9.1). Pygame is a set of Python modules designed for writing games.
Figure 1: Environment. Green is food, Blue is Water, Grey is empty and the circles represent the animats.

- \( Price = \frac{1.0}{\|water - food\| \ast \left(\min(water, food)/100.0\right)} \), if selling food
- \( Price = (\|water - food\| \ast \left(\min(water, food)/100.0\right)) \), if selling water
- \( Quantity = \|water - food\| \ast \frac{Price}{(Price + 1.0)} \)

Figure 2: Price & Quantity Setting
\[ \text{GoToFoodOutput} = \frac{1}{1 + \exp\left( - \frac{1}{\text{FoodSmell} \cdot \text{StoredFood}} \right)} \]

\[ \text{GoToWaterOutput} = \frac{1}{1 + \exp\left( - \frac{1}{\text{WaterSmell} \cdot \text{StoredWater}} \right)} \]

Figure 3: Brain of Gathering Animats
Figure 4: Experiment 1. Prosperity, Survivability, Population state and the environment respectively.

Figure 5: Experiment 2. Prosperity, Survivability, Population state and the environment respectively.
Appendix

The code is in python, and the variable names are chosen so as to be self-explanatory. We have attached the actual python files for readability, but we replicate the important classes here for completeness.

Part 1: Animat Class

class Animat: """"A animat class"""
    water = 0.0
    food = 0.0
    animatid = -1
    #Variables from Gathering Animats
    loc = (0.0, 0.0)
    nodeDiffResources = 0.0
    nodeFoodDistanceSensor = -1.0
    nodeWaterDistanceSensor = -1.0
    gotoWaterNode = -1.0
    gotoFoodNode = -1.0
    Motor = -1.0
    Environment = -1
isAlive = True
# Variables from Trading Animats
threshold = 1.0
nodeMinResources = 0.0
nodeTypeSold = -1
nodePriceAsked = 0.0
nodeQuantityAsked = 0.0
nodePriceOffered = 0.0
nodeQuantityOffered = 0.0
nodePriceTraded = 0.0
nodeQuantityTraded = 0.0
TradeIdOffered = -1
# Variables for the combined environment:
nodeMarketAvgFoodPrice = 0
nodeMarketAvgWaterPrice = 0
# Functions in the Gathering Environment
def UpdateExternalFoodSensor(self):
def UpdateExternalWaterSensor(self):
def UpdateGotoFood(self):
def UpdateGotoWater(self):
def MoveOneStep(self, target):
def Collect(self, _type, key):
def UpdateMotor(self):
# Functions from Trading animats
def UpdateMinResources(self):
def UpdateNodeTypeSold(self):
def UpdateAskingPrice(self):
def UpdateQuantityOffered(self):
def PerformTrade(self):
# Functions for the combined environment:
def ConsumeResource(self):
def CheckIsAlive(self):
def UpdateNodeDiffResources(self):
def UpdateAvgMarketFoodPriceSensor(self):
def UpdateAvgMarketWaterPriceSensor(self):
# Function to update and print Gathering/Trading States:
def GUpdateState(self):
def TUpdateState(self):
def PrintState(self):

Part 2: Environment

# Some important constants:
    WATER_REGENERATION_RATE = 0.0 #1
    FOOD_REGENERATION_RATE = 0.0#1
    INIT_RANDOM = False
#Whether the environment is Controlled or Random
USE_LOCALMARKETS = False #Local or Global Markets
MARKET_RANGE=4 #Range of Local Market

class Environment: SIZE = 20 #Environment size
   noAnimats =20 #Number of Animats
   animatsList=[]
   Habitat = {}
   _habitats = []
   _animats = []
   tradingList=[]
   #Functions :
   def init(self):
   def PrintHabitat(self):
   def RefreshHabitats(self):
   def InitializeTradingRound(self):
   def PerformTradingRound(self,trades): #Simulates the market by trading
   based on the prices set by individual animats
   #Functions that control the timestep at which trading and gathering occurs
   def Tick(self):
   def GTick(self):
   def TTick(self,trades):
   #We use pickle to dump the result at each time step to a text file, which we
   then use for graphs.
   def DumpCurrentState(self,timestep):

Part 3: User Interface Class

This was a crucial, non-trivial piece of code. A good interface was essential
visualize the results of the experiments and it was invaluable when we were
debugging.

#Imports
import pygame, sys from pygame.locals
import * from random
import randint
import numpy as np
import threading
import Environment
import os

#List of Modes:
GATHERONLY = 0
TRADEONLY = 1
GATHERTRADE = 2
MODE = GATHERONLY
#Drawing Functions
def DrawMarket(environment, surface, trades):
def DrawCharts(environment, surface, font):
def drawTerrain(environment, surface):
def drawAnimats(surface, environment, font):

#Main Function

def main():
    #Initialize Environment
    #Initialize Animats
    #For each timestep
    #Call Drawing Functions
    #Update Environment and Animats