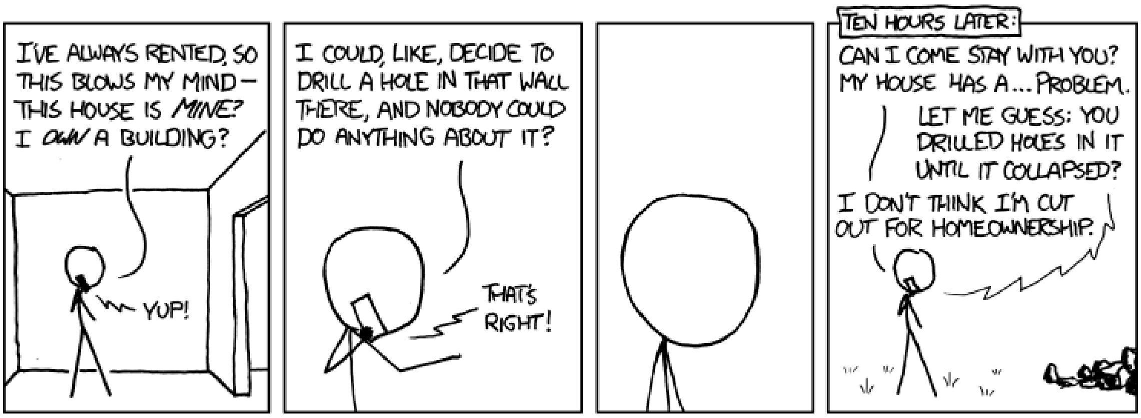
Case Study: Rental Prices

CITS4403 - Computational Modelling Assignment

Mitchell Pomery (21130887)

May 19, 2015


I'VE ALWAYS RENTED, SO THIS BLOWS MY MIND— THIS HOUSE IS MINE? I OWN A BUILDING?
YUP!
I COULD, LIKE, DECIDE TO DRILL A HOLE IN THAT WALL THERE, AND NOBODY COULD DO ANYTHING ABOUT IT?
THAT'S RIGHT!
TEN HOURS LATER:
CAN I COME STAY WITH YOU? MY HOUSE HAS A... PROBLEM.
LET ME GUESS: YOU DRILLED HOLES IN IT UNTIL IT COLLAPSED?
I DON'T THINK I'M CUT OUT FOR HOMEOWNERSHIP.

xkcd.com/905

1 Introduction

Finding a place to live in while renting can be very difficult. Expensive places all seem to be grouped together, right next to each other and all occupied while cheaper places are on the outskirts of town and relatively empty. There are many reasons why this could be, A good way to look at a possibly reason as to why this happens is by using agent based models. These models simulate the behavior of agents in an environment. This allows us to see the overall effect of this behavior on the system.

http://en.wikipedia.org/wiki/Agent-based_model

Simulating population movements using agents is nothing new. Thomas C. Schelling’s ”Dynamic Models of Segregation”, published in 1971, was among the first to do this. His model had two agents that both behaving the same way. They would move if they were ”unhappy” because they were outnumbered by the other agent.

Simulating a city of people moving around to houses with changing rent prices is harder to implement. Several variables such as rental price, occupancy and rent history need to be maintained, and a system needs to be created to determine who is going to move into each house at the end of each month, rather than randomly reassigning them.

2 Implementation

To simulate a city with people moving around it we need to create two agents, the houses themselves and the people who live in them. We then create a city for these houses to be built in and a population that live in this city. These two groups then interact with each other and act depending on the state of the other. At the end of the simulation there are two things we can look at, rental prices and income distribution.

Since the system we are creating is complex, making the code easy to read and understand is important. One way to do this is through the use of pythons named tuples. They allow the creation of unchangeable tuples with properties that can be accessed by name instead of index. They easy to use and make code significantly more readable.

```
import collections

# Define a named tuple
Coordinates = collections.namedtuple(’Coordinates’, ’x y’)

# Create a tuple
location = Coordinates(4, 5)

# Will print 4
print(location.x)

# Will print 5
print(location.y)
```

2.1 Agents

House

- Have a price and possibly an occupant
- Increase rent after certain amount of time
- Decrease rent to try and get people to rent it

People

- Similar to real people
- Have an income
- Have a bank balance
- Pay rent every month
- Can only move when rent agreement finishes
- Unlike real world
- No share houses
- No income increases
- Don't have sentience

2.2 Managing the Agents

City

- Manages all the houses
- Makes sure rent changes

Population

- Manages all the people
- Makes sure they are happy
- Make sure they move if they aren't
- Each step updates rent of old houses, moves people, then updates rent of occupied houses

3 Exercises

Exercise 1: *Randomly give people an income based off of Bureau of Statistic data.*

Exercise 2: *Modify the program to take city size and population size as arguments. See what happens when you increase and decrease the population.*

Exercise 3: *Modify the program so that there is a hole in the center of the City. Experiment with different City and population sizes and see how it changes.*