Environment "Firefighter" Documentation

Author: Shuo Jiang (jiang.shuo@husky.neu.edu)

Introduction

The document introduces an environment for multi agent study as several agents are trying to put out fire on several houses. The code is implemented in python and provided with simple interfaces. The environment is decoupled with learning method so reader can try any algorithms on.

Scenario Description

The task of the environment is explained as below

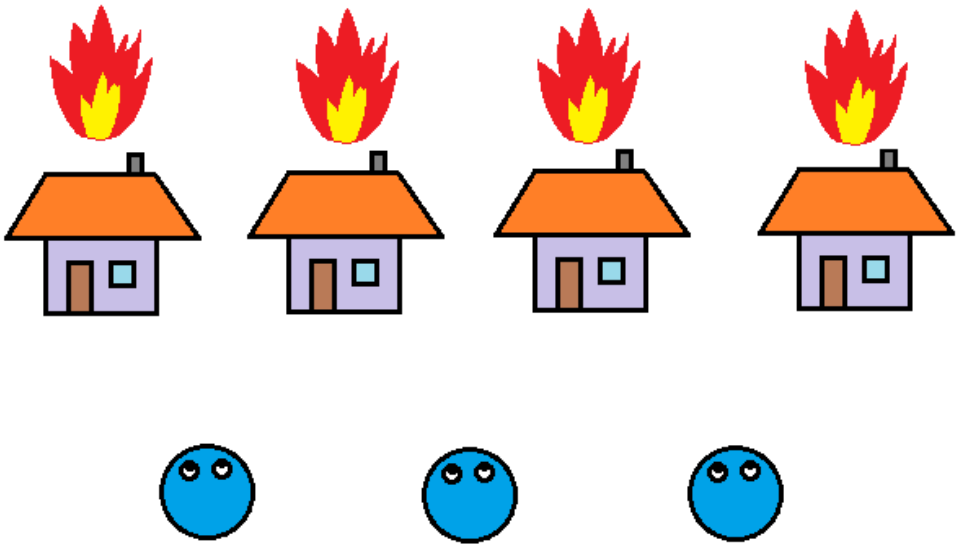

Figure 1

There are several houses on fire. Some firefighters are trying to put out the fire. The number of firefighters is less than the number of houses (specifically less than 1, if there are 4 houses, there will be 3 firefighters). Each firefighter is only able to deal with the two houses near it. As we index the houses as 0-4, and firefighters 0-3, firefighter indexed as 1 can only deal with houses with index 1 and 2.

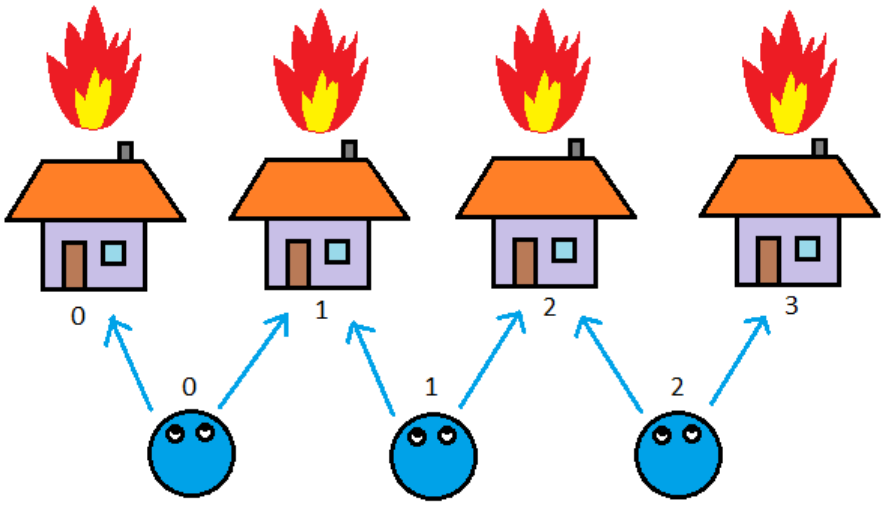

Figure 2

Action Space

At each time step, each firefighter can choose from two actions, "put out fire on the left", or "put out fire on the right". The two actions are denoted by integer 0 and 1. So a valid joint action for 3 firefighters may like: [0, 1, 0]. It means, firefighter 0 puts out house 0, firefighter 1 puts out house 2, firefighter 2 puts out house 3.

Internal Dynamics

So what will happen for the houses on the behaviors of fighters? Each house has a fire level, which is an non-negative integer, 0 means no fire and larger number means stronger fire. The dynamic is stochastic and not necessary to be known to user.

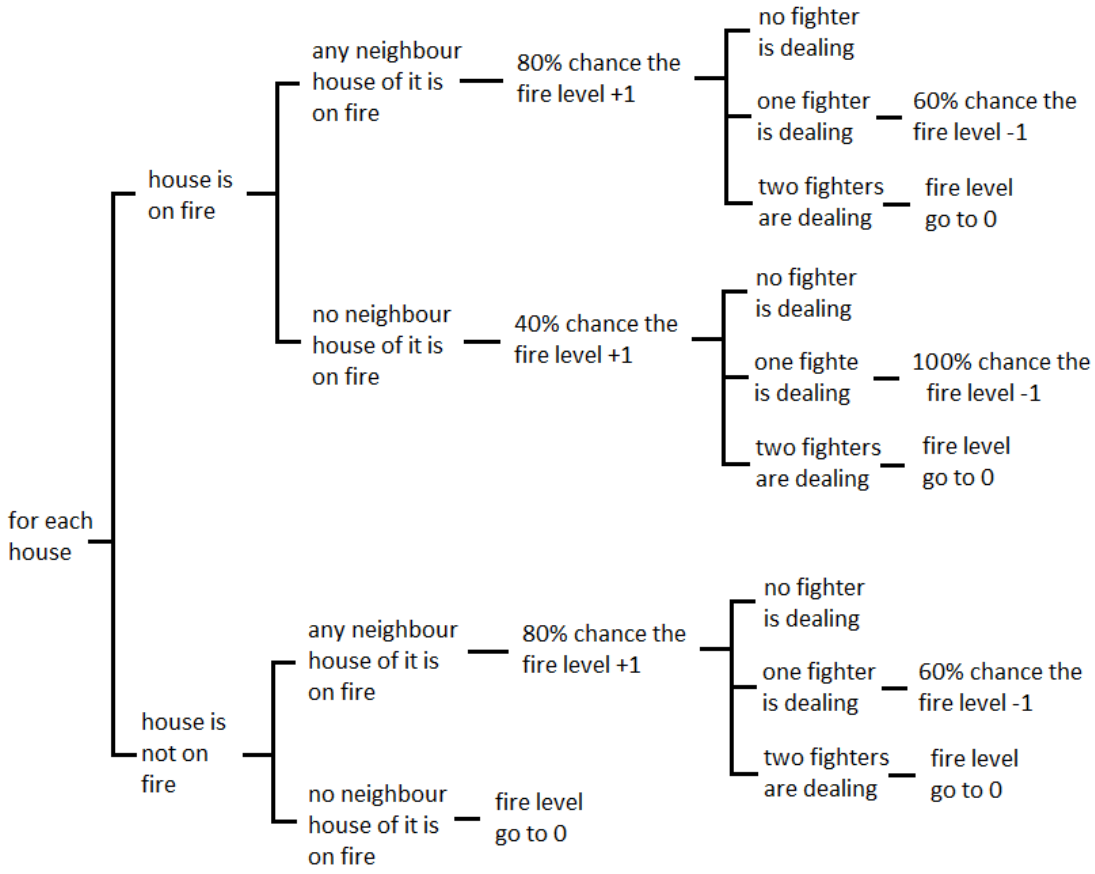

Figure. 3

The reward after each step is the sum of fire levels for all houses. Notice that the first and last house has only one neighbor and can be only dealt by at max 1 fighter.

Observation

The observations of fighters are partial, which means each fighter can only observe the fires of two houses near it. And they cannot observe the fire level but only an integer either 0 or 1 meaning whether there is fire. And the integer is given by chance proportional to fire level. Higher fire level is more likely to be observed as 1 (though it still possible to observe 0). So the observations for 3 fighters may like: [[1, 1], [1, 0], [0, 1]], the real fire level is [12, 2, 0, 14] (This is unobservable).

Class Organization

The environment is programmed in python 3.6 and has very simple interfaces. The file contains the environment is "env_FireFighter.py" and a class " EnvFireFighter" is defined.

Important interfaces are:

Member Variables

```
house_num               # number of houses
fighter_num             # number of fire fighters
firelevel               # list of fire levels for each house, like
                        [12, 2, 0, 14]
```

Member Functions

`__init__( house_num)`
initialize environment with number of house. Initial fire levels are 3 for all houses.

`step(target_list)`
evolve environment with joint action, "target_list" as [0, 1, 0]. 0 is dealing house left, 1 is dealing house right. Return is a integer value of sum of fire levels for all houses.

`reset()`
Initial fire levels are 3 for all houses.

`get_obs()`
The observations of fighters are partial, which means each fighter can only observe the fires of two houses near it. And they cannot observe the fire level but only an integer either 0 or 1 meaning whether there is fire. And the integer is given by chance proportional to fire level. Higher fire level is more likely to be observed as 1 (though it still possible to observe 0). So return observations for 3 fighters may like: [[1, 1], [1, 0], [0, 1]], the real fire level is [12, 2, 0, 14] (This is unobservable).

Example

Here is an example using test function, and it is in "test_FireFighter.py". The joint action is random chosen for each agent, click and run.

```
from env_FireFighter import EnvFireFighter
import random

def generate_tgt_list(agt_num):
    tgt_list = []
    for i in range(agt_num):
        tgt_list.append(random.randint(0, 1))
    return tgt_list


env = EnvFireFighter(4)
```

```
max_iter = 100
for i in range(max_iter):
    print("iter= ", i)
    print("actual fire level: ", env.firelevel)
    print("observed fire level: ", env.get_obs())
    tgt_list = generate_tgt_list(3)
    print("agent target: ", tgt_list)
    reward = env.step(tgt_list)
    print("reward: ", reward)
    print(" ")
```