MODELING COMPLEX SYSTEMS

FINAL PROJECT: AGENT BASED MODEL FOR FOREST FIRE CONTAINMENT

Max Green, David Landay
University of Vermont
Graduate Students, Comp. Systems and Data Sci.

0.1 Abstract

As wildfires threaten Northwest America it becomes increasingly important to design ways to contain a fire soon after its ignition. Given a fire's origin, the environment, and a constrained amount of fire prevention material, we propose a sandbox modeling tool designed to minimize the damage of a wildfire. Fire propagates according to biased stochastic probability function on a 2 dimensional lattice in this agent based model. As the fire spreads, rational artificial agents are tasked with building fire breaks in real time. Given only spatial context, agents are allowed to act on sites in their immediate Von-Neumann neighborhood. By introducing different tune-able parameters,such as topography, fire strength and resource scarcity, users to compare different containment strategies to minimize cost and maximize post fire yields. Our hope is that simulations in this framework may inform real fire fighting efforts after a fire has begun. Ideas are inspired from Highly Optimized Tolerance models and other probabilistic cellular automata models.

0.2 Introduction and Background

In order to optimize resource allocation during extreme wildfire containment efforts, we design a model to test for optimal containment strategies. Fire propagation is the perfect candidate for cellular automata (CA) as it spreads on contact continuously through space. Wildfires have been modeled as CAs many times in the past. Depending on contagion mechanism, CAs have be excellent wild fire modeling tools. Studies have imposed 2D meshes on top of true wildfire image stills and compared real-time and model propagation to varying degrees of success. Some of the most notable fire propagation models include ForeFire, FIRESTAR and FIRETEC proven accurate in fire propagation prediction, however achieving real time results is still a ways off[]. A recent model aimed to predict the Aullene wildfire took a spatial resolution of 24 million grid points and took 9h to complete the simulation. The AUllene fire itself well exceeded this time frame[2]. A huge need for fire propagation models is speed and efficiency to render real time predictions and results.

More complex and layered models require more time to compile and simulate the spread of the fire. Fire spread is influenced by enumerable variables. However, topography, fuel type, wind and humidity are thought to be heavily influential [1].

With this and many previous studies of fire spreading in mind, an agent based model appears to best suite the constraints of the problem. An additional benefit of using an agent based model for this particular problem is the ease of adding additional features to the model at a later date. For example, adding wind vectors would only require an array of velocity vectors to be layered on top of topography arrays. A CA was chosen to model our system of interest for fit,convenience, and maneuverability.

0.3 Methods

0.3.1 Assumptions of Model

As is the case with many models,many assumptions were made in this model. Most prominently, in this iteration, we assume that the environment that the fire takes place in is devoid of wind, an environmental variable known to effect fire spread. We also assume that fire is more likely to spread uphill, this is also typically thought to be true[[1]]. We also assume homogeneous fuel type, all of the trees in the forest are of the same composition and are equally spaced. In the next iteration of this project, lifting these assumptions is prioritized. Additionally, our model has no periodic boundary, when the fire propagates to the border of the grid, it moves no further.

0.3.2 Fire Propogation

The simulation begins on a 2D grid. Every site on the grid has a height, dictated by the landscape, and a state. A sites state can be "Fire","Ash","Tree","Fire Break". All sites are initiated in "tree" state. A random site is selected and set on fire. At every subsequent time step the Van-Nueeman neighborhood[], the orthogonal neighbors of a site that is on fire, are considered for fire spreading. All sites on the border of the fire can feasibly catch fire in the next time step at once. In this way, the fire spreads in a semi-synchronous fashion. We assume that exclusively sites neighboring a fire can catch, i.e there are no sparks that set disconnected sets of trees on fire.

The probability of fire spreading to a site is determined by the following probability function:

$$\gamma + \frac{(1-\gamma)\sum_{i=1}^{k} \Delta\, Z_{ji}\, f_i}{nF_j\ (2\,\mathbf{Max}(\Delta\, Z_j))}$$

where nF_j is the number of neighbors of site j that are on fire, $\Delta Z_{j,i}$ is the difference in surface height between a site j and it's neighbor i, and ΔZ_j is a vector representing each of the differences in surface height between a site j and it's k neighbors. f_i is a term that indicates whether a neighbor i of site j is on fire, and can take on a value of either 0, or 1 as follows:

$$f_i = \begin{cases} 1 & \text{if site } i \text{ is on fire} \\ 0 & \text{otherwise} \end{cases}$$

The term γ is a tune-able parameter that determines the strength of the fire, independent of the spatial dependencies described above. One may think of γ as the likelihood of a site burning on a plane surface. The graphic below acts a visual aid to show how the likelihood of a site catching fire is determined by the relative heights of it's neighbors. Additional features of the landscape and environment could easily be added to this function.

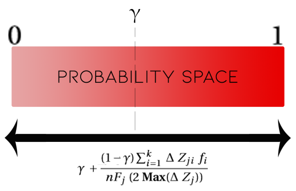

γ
0
1
PROBABILITY SPACE
$\gamma + \frac{(1-\gamma)\sum_{i=1}^{k} \Delta Z_{ji} f_i}{nF_j \, (2\,\mathbf{Max}(\Delta Z_j))}$

Figure 1: γ is a tune-able parameter that dictates the likelihood of a site catching fire, independent of spatial information.

0.3.3 Agent Logic

Rational agents are tasked with enclosing the fire as it spreads over a landscape. The agents operate under a simple rule set. In our model, agents are not aware of the underlying probability distribution that dictates the spread of fire over the state space at any given time. They are also confined to a finite a of amount fire break material per update. Agents are placed on the landscape next to areas on the map are most likely to catch fire in the next time step. The placement strategy is aimed to emulate role of smoke jumpers", wild fire fighters that are dropped out of the air via helicopter or airplane and land directly into or near the border[3]. In order to distribute the efforts of the agents across the border of the fire, the landscape is partitioned by number of agents. Each agent is assigned to a partition and will be placed within it via the procedure described above.

After the agents are placed, at each time step they must update their position.At the same time, a spatial risk map is calculated based on the shortest path between every site and a site that is on fire. There are a few exceptions to this. Sites that are currently on fire have a risk of zero, along with sites that are ash and trees detached from the giant component. At a given time step, agents choose a neighboring site which has the highest spatial risk. This way the agent is drawn towards the fire but will never run into it. The number of spaces the agent can move in a turn are given as a tune-able testing parameter. Whenever an agent moves a space, a fire break is made. The agent decision making algorithm goes like this:

```
∗ For number of allotted blocks per turn:            (1)
        Scan Neighbors                               (2)
        Move to neighbor with maximum spatial risk   (3)
        Place fire break                             (4)
```

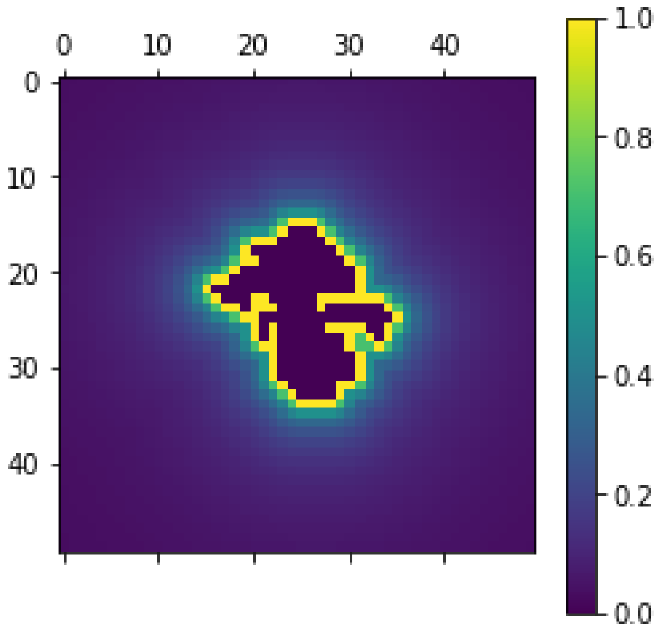


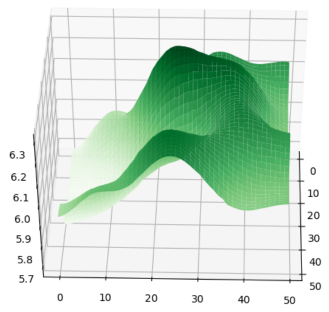
Figure 2: Spatial risk map calculated at the time after fire outbreak. Lighter colors indicate areas of high risk, while darker sites represent low risk areas. Agents will calculate the shortest distance to the site of highest risk and place the allowable number of fire breaks in the direction of that site

This way, agents will attempt to traverse the boarder of the fire and enclose it with fire breaks. The success of the agents is varies highly with the amount of resource they are given and the initial strength of the fire.

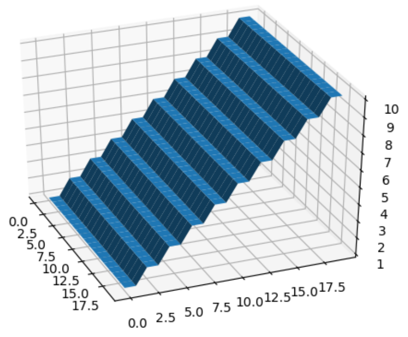
0.3.4 Generating Terrain:

In real world situations, wild fires can spread for many miles, over a variety of different terrains and topographies. The amount of landscape diversity in regions that are prone to wild fire adds uncertainty when attempting to create optimal strategies to combat their spread. It is clear that the state space of our model is quite expansive in that varying terrains could greatly impact the success or failure of fire containment. Hence, we would like to explore optimal containment strategies over a sweep of different terrain types.

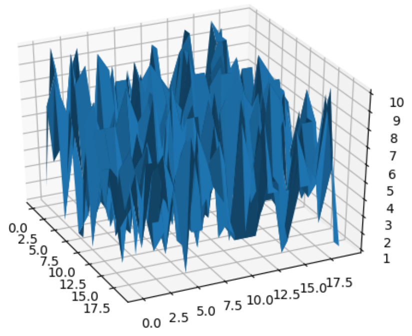
To observe the effect that landscape has on the optimization techniques employed by the agents, we generate smoothed-random surfaces to simulate fire containment on different terrain types. We implement Perlin noise[3], an image synthesis technique, to create 2D height maps. By assigning random height values chosen from different Gaussian distributions over a 2D lattice, we can interpolate slopes between sites and their neighbors to describe smooth gradients over the embedded noise. As we interpolate in time, we may achieve smoother and smoother gradient maps. From [4], we use the quintic fade function $6t^5 - 15t^4 + 10t^3$ to interpolate smooth curves between all sites and those within their Von-Neumann neighborhood. The function maps discrete coordinates from the Cartesian plane onto a surface that we use to represent a landscape. **Fig.1** shows an example of a height map that this technique can produce, as well as other synthetic topographies that we can employ in our model. **Fig.2** shows our implementation of the Perlin noise, image synthesis technique.



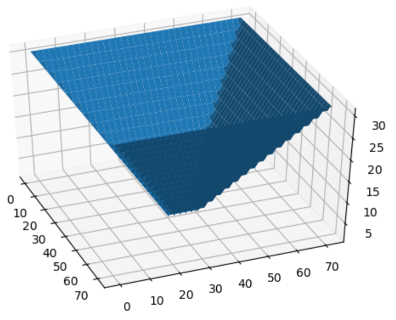
(a) Smooth gradient surface generated with Perlin noise, interpolated over 100 timesteps



(b) Tilted plane (steps), synthetic ramp surface

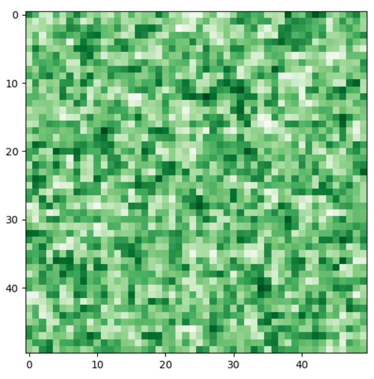


(c) Surface generated with random noise (micro-scale topography)

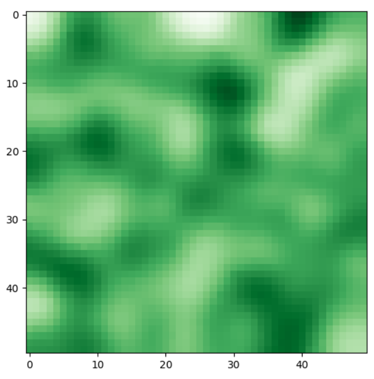


(d) Synthetic bowl surface topography

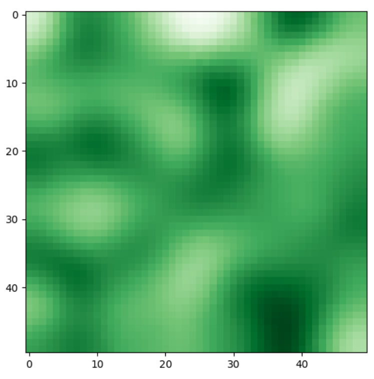
Figure 3: Examples of different surface gradients, over which we simulate wild fire propagation



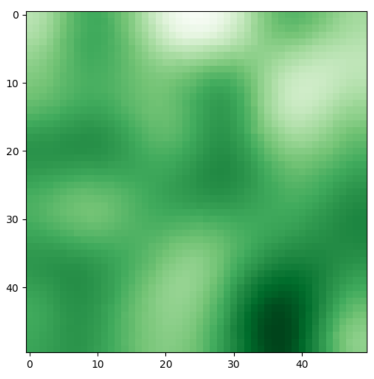
(a) Random noise, t_0



(b) Perlin Noise, t_{24}



(c) Perlin noise, t_{49}



(d) Perlin noise, t_{99}

Figure 4: Examples gradient smoothing over 100 time steps

0.4 Results & Conclusion

To assess the quality of success of an agent strategy given a varied number of resources per time step, we fixed a landscape and tracked the forest yields over each landscape post fire for 50 trial runs. Total yield is taken to be the number of trees remaining after an outbreak. The figure below shows the maximum yields for a synthetic bowl-like topography, and a height map generated from Perlin noise. Since our agents cannot assess the quality of the placements of their resources, containment remains a loosely defined construct. For example, it is likely that extremely low yield sizes, that are not equal to zero, indicate that agents were able to section off a portion of the forest very soon after a fire outbreak, but ultimately failed to prevent the fire from burning all of the trees. In some cases, the agents could not determine an initial starting position, based on the initial conditions of the fire. Thus, some simulations threw `nan` values. In either case, we had to consider containment as a binary problem, in which a simulation results in a success as long as there exists remaining yield. In place of calculating the mean yield for all 50 trials, we penalize the maximum yield over all simulations for each of the different resource quantities. The penalized maximum yield for a strategy after each trial, given b resources, then becomes

$$y_b = \mathbf{max}(Y_b) \times \frac{F_b}{N_b - n_b}$$

where Y_b is a vector containing the yield values for each trial run, given a specific topography and b number of resources, and F_b is the number of failed trials over the simulation, normalized by the difference in total number of trials N_b and total number of `nan` values that appear in the simulation.

We see that initially, the yield increases linearly with the number of allowable resources. But at a certain point, the gains diminish and actually reduce the yields by unnecessarily replacing trees with fire breaks

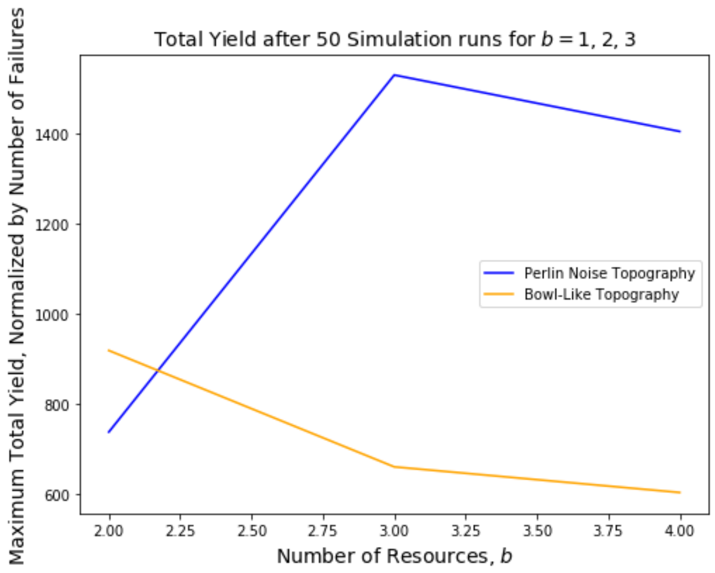


Figure 5: Comparision of Strategy Over two topography types, given variable resource allowances and with fixed fire spreading rate$\gamma = 0.2$

The results, while not extensive, indicate that the same strategy can fail to contain a fire more reliably when placed on different types of terrains.

0.5 Future Work

The sandbox nature of our model allows us to compare different containment strategies, and design more advanced agent logic. It is likely that optimization strategies will outperform others in different scenarios; for instance, over specific topographies, or under variable wind conditions. We would like to explore, and compare, these strategies to know how robust they are to their environment.

Aside from implementing and comparing additional agent strategies, like intelligent reinforcement learning schemes, there are various improvements that we would like to make to our sandbox model. We hope to expand upon the capabilities of our framework, by creating a tool to inform policy makers in the event of a wildfire. With the advent of openly available geospatial and Light Imaging Detecting and Ranging (LIDAR) datasets, we are no longer limited to using synthetic surface maps in our simulations. Our aim is to model optimal fire-fighting strategies, based on specific geographic conditions, as a way to minimize resources used and risk taken when fighting actual forest fires. We see potential for real-time analysis and decision making, given up-to-date surveys of a prevailing outbreak.

Bibliography

[1] Fire Behavior Variables.

[2] Jean-Baptiste Filippi, Frédéric Bosseur, Céline Mari, and Christine Lac. Simulation of a Large Wildfire in a Coupled Fire-Atmosphere Model. *Atmosphere*, 9(6):218, June 2018.

[3] Ken Perlin. An Image Synthesizer. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’85, pages 287–296, New York, NY, USA, 1985. ACM.

[4] Ken Perlin. Improving Noise. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’02, pages 681–682, New York, NY, USA, 2002. ACM.