Mosquito Writeup

Group 4: Hari Rajaraman, Sam Raper, Patrick Wingo

CIS 700

Introduction 3

Initial Insights and Observations 3

Strategies and Concepts 5

Implementation 6

Results 6

Contributions 6

Future Directions and Limitations 6

Acknowledgements 6

Conclusion 6

# 

# Introduction

In this problem, we were given a 100x100 board filled with mosquitos that move randomly. Our goal was to place and move lights that “capture” these mosquitos and bring it to a “collector” as quickly as possible. The board can also have any number of “walls” that will prevent a light from crossing.

We started out with three different implementations on a blank board to get the feel for the problem. We describe them here and will discuss what we learned in the very next section:

**Approach 1:** *Sweep from top to bottom.* We docked the collector near the point (50,98), right about the bottom center and placed the lights at the top. They were spread out such that they collectively ‘cover’ the board’s width. From here we move the lights in a vertical fashion until the y coordinate is 98. Then we move it horizontally towards the collector.

**Approach 2:** *Sweep from top to bottom with oscillation.* This approach was effective when the lights didn’t cover the board’s width. If we were to just move it straight down, it would leave out all mosquitos in the uncovered areas of the board. To combat this, we would bring the lights down 40 units and then move it towards the edge of the board and then down 40, then towards the initial x coordinate and form a zigzag motion. The specific behavior is shown in Figure 1

//TODO: need a figure with Patrick’s screenshot

**Approach 3:** *Divide board into section and collect mosquitos from each section.* We split the board into 16 sections of 25x25 units, small enough for one light to cover the entire section, and placed the lights in the most ‘dense’ sections. Each light will go to the collector, and then proceed to next dense section that’s still remaining. We also had one light docked at the collector so that any mosquitos near the collector won’t get lost. When a light leaves from the collector to the next section, it will turn itself off in order to prevent dragging any of the mosquitos away from the collector.

# Initial Insights and Observations

We made some key observations when watching our players in motion. Some of them are very trivial and almost obvious while other takeaways were critical to our future implementations. We’ll list them all for the sake of completeness:

*A light point doesn’t need to cover the entire board.* To capture a mosquito, the only requirement is that radius around that light hits all the areas of the board and any mosquitos inside that radius will travel towards the light. While this is pretty obvious, we can extend this to say:

*The closer a mosquito is to the center of the light, the lesser chance that the light will “lose” it.* This is true when a light moves around obstacles and makes a “tight” turn around a wall. If any mosquitos are near the edge of the circle, they will no longer be in the “line of sight” of the light and as a result, will no longer follow it. Even one lost mosquito means another light needs to collect it in the future.

Through these two points, we realized that a “sweeping” strategy is essential. We call an area as “swept” if a light moving through it has touched every point and has a direct light of light with that point. This means that any mosquitos in that point will promptly move towards the light and will be captured.

*Creating simple quadrants made the sweeping algorithm very simple.* In our third approach, the algorithm was simple: go to center. The sections were so small that the radius around the light easily encompassed the 25x25 grid. As soon we covered the section we can be sure that the mosquitos will travel towards the light. Even if the light starts to move away from the section, we know that the mosquitos will follow. It’s also important to note that we could even get away with not going fully to the center a few units inside of that section would cover everything.

*Collecting the denser sections first resulted in a quicker runtime.* Imagine a scenario where a light in board A drops off a 100 mosquitos and 10 mosquitos during its second trip; and in board B a light drops off 10, then a 100. Assuming that the distances traveled to the sections are the same in both boards, the first board would finish first. This is due to the fact that while the lights were out for their second trip, the collectors will have 100 and 10 mosquitos to collect and chances are they would finish collecting before those lights return. When that happens, board A’s collector will only have 10 lights to collect whereas board B will now have 100. Given the randomness of the mosquitos’ movements, it’s more likely that those 10 mosquitos will be collected first.

While this was nice to know, it really didn’t help us in the long run. There were so many assumptions made in that scenario, specifically section location and distances, that the chances of that happening were very low. We also realized creating some sort of algorithm from this would yield us low returns.

We also observed certain things that our implementations didn’t do well.

*Our final score depends on the last light that reaches the collector.* In fact, our score is the number of rounds that light takes to finish plus some rounds for collecting. One of our classmates put it more eloquently: “minimize the longest trip”. This hinted as some sort of collapsing algorithm where each light would travel a lesser distance. Of course, even if a few lights take a longer route, we will improve our score as long as the longest light takes the shorter route. A prime example in the blank board would be to place the collector in the center and spread out each light and then bring each of them towards the center.

*Only go to the collector when needed.* Our third implementation would sweep a quadrant, then repeat until all quadrants are swept. This was particularly inefficient as the light makes an unnecessary trip to ‘drop off’ lights whereas sweeping the nearest section would save two trips to the collector – one to drop off and another to reach the section.

*If above statement is true, then it is useless to dock a light at the collector.* The solepurpose of that light is to ensure dropped off mosquitos don’t wander off and it’s no longer needed if we no mosquitos are dropped off.

# Strategies and Concepts

From the above two insights we figured out that an efficient sweeping strategy is very critical to our performance. A light should cover the entire area of whatever shape/section/quadrant it is given with the minimum number of moves. Along with this idea came a few questions: what exactly is a section and what should it look like? Can we somehow classify a partition as “sweepable” versus “non-sweepable”? Assuming we can make a great sweeper, how can we partition the board to take advantage? The third approach answered a lot of those questions.

*Our sectioning algorithm should produce the simplest partitions.*  We defined a partition (or section) as simply a contiguous set of points within the board. We’ll call a partition simple if:

* It doesn’t have any obstacles
* A light traversing this section should cross this

# Implementation

# Results

# Contributions

# Future Directions and Limitations

# Acknowledgements

# Conclusion