Distributed Channel Assignment Using a Modified ALOHA Protocol

Robert J. Vasil

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2015

Committee:

Mehran Mesbahi, Chair

Kristi Morgansen

Program Authorized to Offer Degree:
Aeronautics and Astronautics

University of Washington

Abstract

Distributed Channel Assignment Using
a Modified ALOHA Protocol

Robert J. Vasil

Chair of the Supervisory Committee:
Mehran Mesbahi
Dept. of Aeronautics and Astronautics

In this thesis, we propose a distributed channel assignment algorithm for multi-agent robotic systems using infrared LEDs for local positioning and communication. Specifically, we look at a modification of the ALOHA Medium Access Control protocol which is then tested on static and mobile agents under various dynamics. This positioning and communication system is similar to one proposed in [13]. The purpose of the local positioning and communication system is to reduce the infrastructure required for experiments involving robotic swarms and to increase the number of agents that can be involved in these experiments.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to thank Dr. Morgansen for being a part of my reading committee. I would also like to thank Dr. Mesbahi for his guidance and input while researching and writing this thesis. Lastly, all simulations for this thesis were conducted using GNU Octave[1] and Processing[2] open-source, mathematical and visualization software. I am a firm believer in the open-source community and feel that the open-source ethos most closely aligns with the scientific community's ideals for the advancement of knowledge and the betterment of society.

[1]https://www.gnu.org/software/octave/

[2]https://processing.org/

Chapter 1
INTRODUCTION

1.1 Local Positioning and Communication

Typical experiments involving multi-agent robotic systems utilize motion tracking and wireless communication systems to coordinate the dynamics of the robots (agents). Although the agents may act autonomously, they often require a motion tracking system for spatial awareness. This motion tracking system captures the locations of each agent which is then broadcasted over the network. Since the agents must constantly be told where they are spatially, there is a considerable demand on the communication network, and this demand limits the number of agents that can be involved in any one experiment - about a dozen agents in the case of our lab.[1] Furthermore, since agents typically communicate over WiFI, they have the ability to communicate to all other agents despite their proximity; this is due to the relatively long-range abilities of WiFI compared to the relatively small area that is typically covered by experiments. And although a complete communication network is often desirable from a dynamics perspective, it is not representative of the distributed systems that are the focus of most experiments.

Currently, our lab is working on a simple local positioning and communication system using infrared LEDs that will allow us to test distributed dynamics that more closely resemble real world applications. Infrared (IR) was chosen as the communication medium due to its relatively short range capabilities compared to other media such as RF and due to its low cost and power requirements. Utilizing this network, agents triangulate their positions against other agents and known 'landmarks' (can also be immobile agents) for spatial

[1]The Robotics Aerospace and Information Networks lab: http://rain.aa.washington.edu/

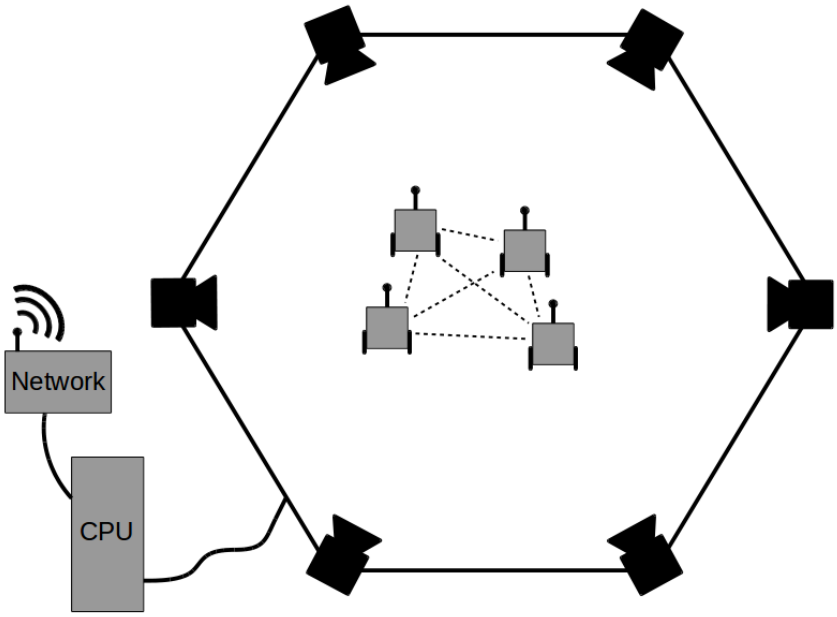

Figure 1.1: Typical infrastructure required for multi-agent robotic experiments. This system includes: several motion tracking cameras, a central processing unit for the motion tracking system, and a network for broadcasting agents locations and neighbors. Dashed lines represent the communication network of agents. Due to the requirement of a network router, the number of agents capable of being in a given experiment are limited.

awareness. As a result, the large demand on the communication network is lifted because the agents positions are not broadcasted over the network. This increases the number of agents that can be involved in any one experiment. Secondly, this local network can be used for communication among agents. Specifically, we suggest using this communication network to determine which agents communicate over WiFI, artificially creating a distributed WiFI network. Throughout the rest of this thesis, we will refer to this local positioning and communication system as the Beacon system. And, one of the main advantages of the

Beacon system is that it relieves the researcher of the considerable infrastructure required for experiments, allowing them to showcase their research contributions by making virtually any setting available to them with only a few 'landmarks' needed for local positioning.

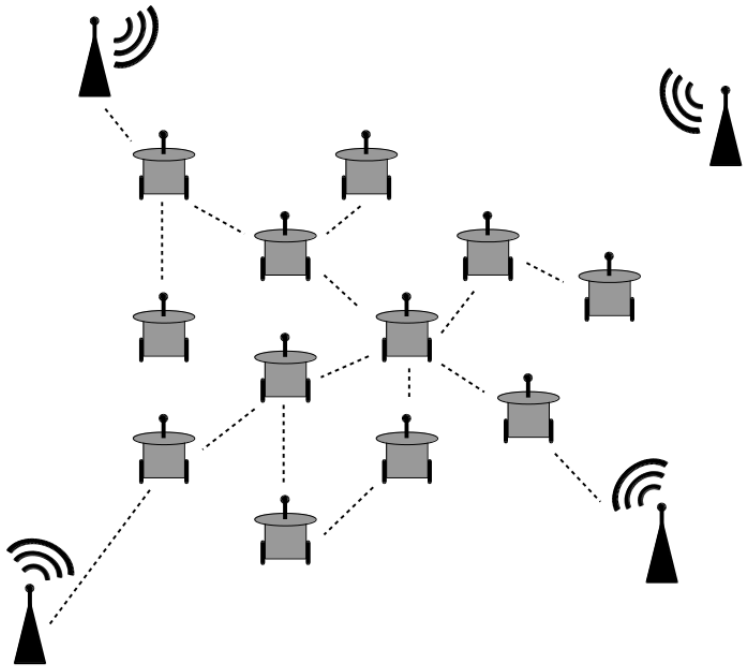

Figure 1.2: Without the requirement for a motion tracking system the network router is also not required making it possible to conduct experiments with more agents than previously possible. Agents can also choose to only communicate with their neighbors if desired. Landmarks (black triangles with antennas) can be used for localization.

1.2 Sensing and Communication System

In [13], a similar low cost communication and sensing system utilizing IR has been developed and tested for experimenting with microrobot swarms. In this article, they discuss the

limitations of the 802.11 standard for micro-controllers communicating over IR and suggest a 6-channel, directional communication protocol to alleviate the problems of medium access. Utilizing this setup, the sensing and communication disk around each robot is divided into 6 equal sectors with a single channel devoted to each sector. Figure 1.3 illustrates this concept. The purpose of dividing the disk into sectors is to limit the number of neighboring robots that can occupy a sector at any given time. The authors found that sectors between approximately 40° and 60° of separation sufficiently limited the number of robots in the 'close' and 'near' ranges, preventing interference from multiple robots occupying the same communication channel. The novelty of this setup is that the robots do not need a channel assignment protocol because only a single neighboring robot is potentially communicating within each sector. Unfortunately, this sensing and positioning system is intended for microscale applications and an attempt to scale this system up to suit our needs would require significantly more channels.[2]

Due to the limitations of this microrobot sensing and communication system, we instead propose a channel assignment algorithm for the Beacon system to alleviate medium access. The remainder of this thesis is broken up as such: Chapter 2 explains the problem of allocating channels to a multi-agent robotic system. In Chapter 3, relevant work is summarized and the limitations of the methods used are explained. In Chapter 4, we explain the channel assignment algorithm and provide an analysis, and Chapter 5 contains avenues for future research and concluding remarks.

[2]Increasing the sensing distance while limiting the number of robots in a given sector would require narrowing the channel sectors and consequently increasing the number of channels. Even reducing the sector widths to 30°, which is just below the lower end of the authors' suggested range of 40° for the microrobots, would require twice as many channels.

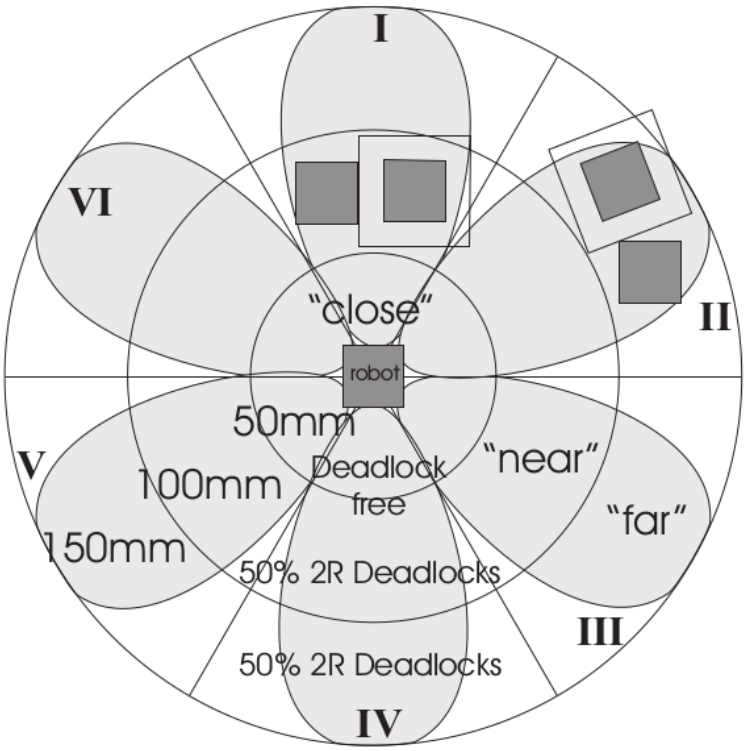

Figure 1.3: Roman numerals indicate channels of sectors and neighboring robots are indicated with dark gray squares. Lighter gray regions indicate the extant of IR capabilities for the particular emitter/collector pair of each section. (Kornienko et al. 2005)

1.3 Notation

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denotes the undirected graph composed of the vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. $\tilde{\mathcal{G}} = (\mathcal{V}, \tilde{\mathcal{E}})$, where an edge $(i, j) \in \tilde{\mathcal{E}}$ if $(i, j) \in \mathcal{E}$ or, inclusive, if nodes $i, j \in \mathcal{V}$ have a common neighbor, i.e., $\tilde{\mathcal{G}}$ is the union of $\mathcal{G}$ and the two-hop graph of $\mathcal{G}$. The line graph of $\tilde{\mathcal{G}}$ is denoted by ${}^{l}\tilde{\mathcal{G}}$. $\mathcal{N}(i)$ and $\tilde{\mathcal{N}}(i)$ are the neighbors of node i in $\mathcal{G}$ and $\tilde{\mathcal{G}}$, respectively, and ${}^{l}\tilde{\mathcal{N}}(g)$ is the neighborhood of edge g in the line graph of ${}^{l}\tilde{\mathcal{G}}$. Δ and $\tilde{\Delta}$ are the maximum degrees of $\mathcal{G}$ and $\tilde{\mathcal{G}}$, and $c(i)$ denotes the color of vertex i. The probability that a node i is the same color as node j is $P_i(j)$. $P_i(j \cap l)$ denotes the probability that a node i is the same color as both nodes j and l, and $P_i(j \cup l)$ denotes the probability that a node i is the same

color as either nodes j or l, inclusive. Text in typewriter font symbolizes pseudo computer code or parameters passed between nodes, e.g. `parameterPassed`.

Chapter 2

PROBLEM FORMULATION

2.1 Inter Vehicle Communication

The Beacon system is designed to work by individual agents broadcasting their unique Vehicle Identification Number (VIN) and location[1] via infrared LEDs located on the perimeter of the agent. This is similar to [13] except that there is only a single channel sector now spanning the entire sensing disk. Sensors of neighboring agents detect this unique VIN and initiate communication over WiFI with this agent. A communication link between agents is established when the agent that receives a WiFI request sends a WiFI confirmation to the original agent; this is often referred to as a handshake. Advantages of this system over the current method of communication is that WiFI can still be utilized to transmit large amounts of data[2], while artificially creating a distributed communication graph, and 'landmarks' can utilize these same Beacons, giving agents references to localize themselves to within the geographical space.

As with any communications medium, there is a limited number of channels that the agents can broadcast on without causing interference for neighboring agents, i.e., if two agents are communicating on the same channel and they are both adjacent to a third agent their signals will interfere with each other, preventing the third agent from initiating communication with either of them. The reason for the interference is because the IR collector of

[1]For landmarks this location is fixed, but for mobile agents their known location is found through consensus and triangulation with their neighbors.

[2]The Beacon system can still be used for communication, but initial estimates signify that the transmission speed is limited to about 1 kB/s.

the third agent will pick up the superposition of the two individual signals.[3] Then, the goal of this thesis is to present a distributed method for assigning channels to agents utilizing the Beacon system.[4] Furthermore, we attempt to quantify the communications success for mobile agents utilizing this protocol under various dynamics; Brownian motion was found to be particularly useful since it exemplifies the types of interactions that are possible among multi-agent robotic systems, including worst case scenarios.

2.2 Graph Coloring

The problem of assigning channels is often abstracted to coloring the nodes of the communication graph - the method we have adopted for this thesis. In the graph coloring problem, each communication channel is represented as a color and agents are represented as nodes. For the purpose of this thesis, edges in $\mathcal{G}$ represent that agents are within sensing distance of each other but not necessarily that they have formed a communication link over WiFI. A communication link can only be formed when agents are within sensing distance of each other and the graph $\tilde{\mathcal{G}}$ is *properly colored*.

Definition 1 (Properly Colored) *A graph $\tilde{\mathcal{G}}$ is properly colored if, for all $i \in \mathcal{V}$ and for all $j \in \tilde{\mathcal{N}}(i)$, $c(i) \neq c(j)$.*

Similarly, agents will lose communication links once $\tilde{\mathcal{G}}$ is no longer properly colored. Figure 2.1 shows examples of improperly and properly colored graphs. It is important to note that

[3]Processing a signal that is the superposition of two separate signals is possible; the new signal is simply the sum of the individual signals. However, this becomes more difficult when multiple signals are added together with varying amplitudes and frequencies, within the processing bandwidth. And because we are using microcontrollers with limited computing resources, it is easier to designate separate channels to agents than trying to reconstruct multiple signals.

[4]It should be noted that, in this thesis, we are strictly presenting a channel allocation algorithm as apposed to considering time division algorithms for channels. The reason we are not considering time division algorithms at this time, is that our application for this protocol is mobile agents and we want to minimize the amount of time that agents are not in communication with their neighbors.

we are requiring $\tilde{\mathcal{G}}$, i.e., the two-hop graph of $\mathcal{G}$, to be properly colored in order to solve the hidden node problem[5] associated with Medium Access Control (MAC) protocols.

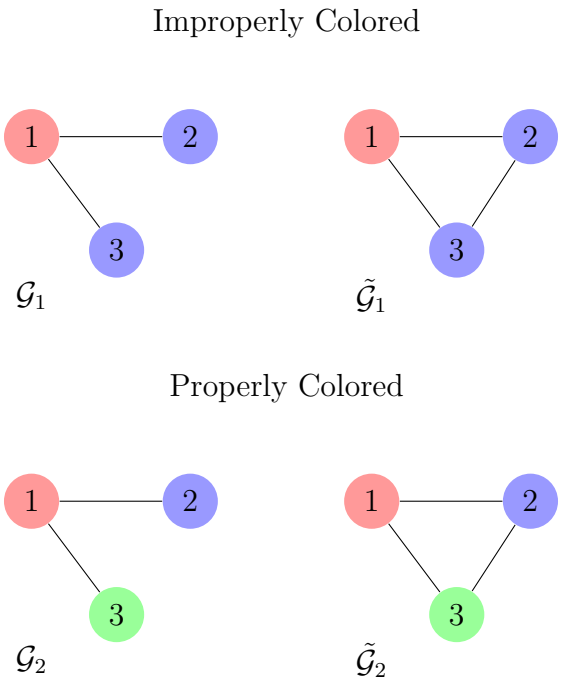

Figure 2.1: $\tilde{\mathcal{G}}_1$ is improperly colored because $c(2) = c(3)$ and $(2,3) \in \tilde{\mathcal{E}}$, but $\mathcal{G}_2$ is properly colored because each node has a unique color.

Due to the interference of signals previously described, it is not possible to properly assign channels to agents with more neighbors than allowable channels - i.e., an agent with 11 neighbors will always have interference if there are only 10 allowable channels - we will only consider graphs where the maximum degree, Δ, is one less than the maximum number of channels, k, for the analytic portions of this thesis. During simulations evaluating various types of dynamics, this restriction is removed, however.

Before continuing with channel assignment algorithms, it is important to understand the relative certainty of conflicts arising if the agents were just randomly assigned channels -

[5]This was discussed in the previous section when we explained the superposition of signals. Two agents that do not know they share a common neighbor are 'hidden' from each other and can interfere with each other's communication with the common neighbor.

this would be a naive attempt at channel deconfliction by having agents randomly assign themselves channels on initialization. From [16], we have a way to analyze this very scenario using the probability and expected value of channel conflicts. Restricting their analysis to hypergraphs with edges connecting 2 nodes, if we have a graph with n nodes, then there are at most $\binom{n}{2}$ edges. Define the random variable $\mathcal{X}_e$ to be 1 if the edge e is monochromatic (i.e., both nodes connected by the edge have the same color), and 0 otherwise. Then, the number of monochromatic edges in $\tilde{\mathcal{G}}$ is $\mathcal{X} = \sum_{e \in \tilde{\mathcal{E}}} \mathcal{X}_e$. Since all channels have an equal probability of being selected and agents select channels independently, the probability of any two agents, i and j selecting the same channel is,

$$P_i(j) = \sum_{1}^{k} \frac{1}{k^2} = \frac{1}{k}, \tag{2.1}$$

where k is the number of possible channels. Then, the expected value of $\mathcal{X}$ is,

$$\mathbf{E}(\mathcal{X}) = \sum_{e \in \tilde{\mathcal{E}}} \mathbf{E}(\mathcal{X}_e) \leq \frac{\binom{n}{2}}{k}, \tag{2.2}$$

since there are at most $\binom{n}{2}$ edges.

The usefulness of this approach involves applying the Markov Inequality for $\mathbf{E}(\mathcal{X}) < 1$. When $\mathbf{E}(\mathcal{X}) < 1$, $\mathbf{Pr}(\mathcal{X} = 0) > 0$, i.e., there is a positive probability that the graph is properly colored after initialization. This means that if $\frac{\binom{n}{2}}{k} < 1$, we can say that there is a chance that there is not a channel conflict after initialization; this is interesting to note since the agents randomly picked channels. Unfortunately, in order for $\mathbf{E}(\mathcal{X}) < 1$, we require more channels than possible edges. This is not practical, but a more useful bound can be found using the Lovász Local Lemma. [16]

Lovász Local Lemma *For events $A \in \mathbb{E}$ with $\boldsymbol{Pr}(A) \leq p < 1$ and each A is mutually independent of all but at most d of the other events in $\mathbb{E}$, if $ep(d+1) < 1$ where $e = 2.7182\ldots$, then with positive probability, none of the events in $\mathbb{E}$ occur.*

The events, A, that we are interested in are whether an edge is monochromatic or not. It is shown in [16] that for edges g and f, the events A_g and A_f are mutually independent if $f \notin {}^{l}\tilde{\mathcal{N}}(g)$, i.e., if the edge f is not in the neighborhood of g in the line graph of $\tilde{\mathcal{G}}$, which is at most $2\Delta - 1$ dependent events in our case. Therefore, there is a positive probability of a proper coloring after initialization if $e2\Delta < k$. What is useful about this bound, is its independence of the number of agents and only restricts the number of neighbors for any one agent. This lemma can be easily tested as Figure 2.2 illustrates.[6]

Testing the lemma, random graphs were generated and the maximum degree of $\mathcal{G}$ and consequently the number of necessary channels was found. Then, nodes were assigned a uniformly random channel and the graph was checked for proper coloring. This was repeated for 40,000 iterations and the frequency of successful colorings can be seen in the figure. This test was repeated on multiple random graphs and always produced a positive frequency of initial proper colorings, although as the figure shows it is a small frequency. This lemma is useful in the fact that it provides insight into the probability of an ad-hoc sensor network or swarm of robots having a proper channel assignment upon initialization and the number of channels that are needed to satisfy this lemma; namely, that it is extremely unlikely and the number of channels that are required for even a small positive probability are unrealistic. Therefore, due to the relative certainty of channel conflicts upon initialization, we need to develop a channel assignment algorithm to take care of medium access.

[6]The ceiling of $e2\Delta$, denoted by $\lceil e2\Delta \rceil$ is used for simulations.

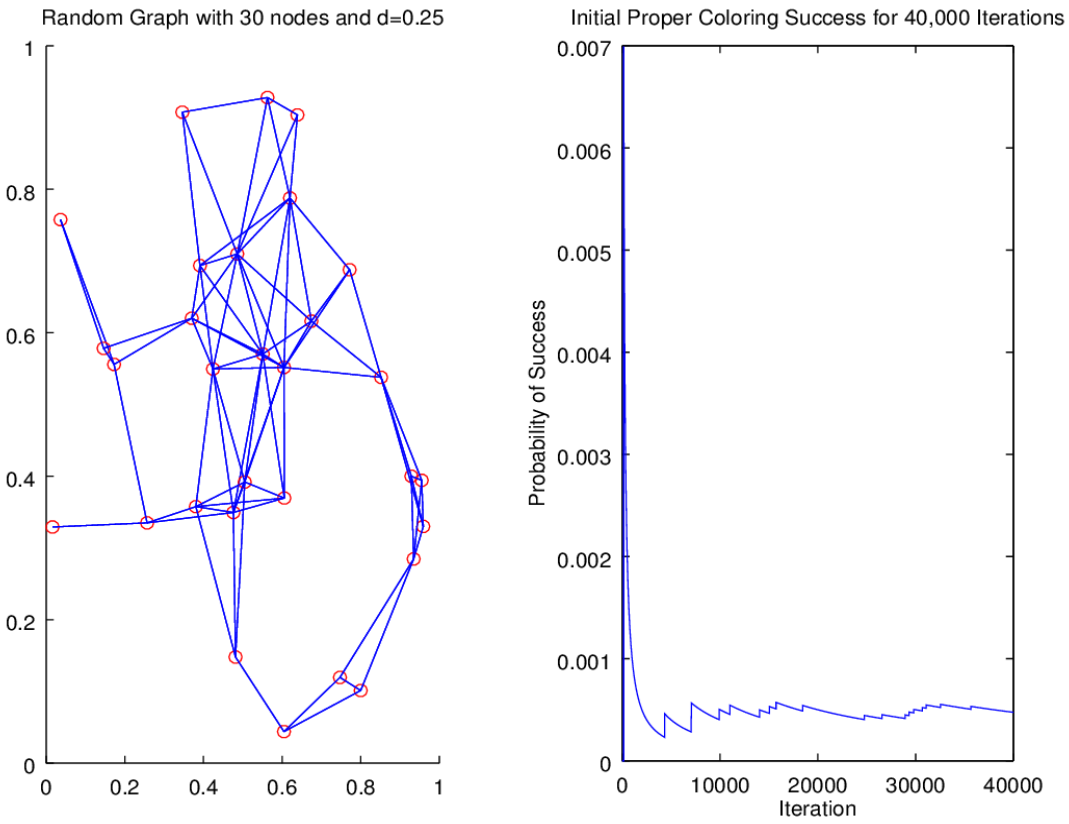

Figure 2.2: Results from testing the *Lovász Local Lemma* on a random graph with 30 nodes and $\lceil e2\Delta \rceil = 60$ available channels in Octave. The left figure shows the random graph, $\mathcal{G}$, the lemma was tested on and the right figure shows the frequency of successful initial proper colorings.

Chapter 3

RELATED WORKS

There are two types of coloring algorithms that we are interested in studying for graph coloring and two subdivisions within these, centralized and distributed coloring methods with the subdivisions being deterministic and probabilistic. By deterministic, we mean that given some initial graph coloring, the algorithm will always lead to the same proper coloring, and by probabilistic, we mean that certain events happen with some probability. The method we propose is a probabilistic, distributed method, but for completeness we feel that it is necessary to discuss both centralized and distributed methods.

3.1 Centralized Coloring

Utilizing a centralized approach, a single agent/computer determines which channels get assigned to all the nodes. This can sometimes be advantageous because the central controller is aware of the entire communication graph. Deterministic, centralized algorithms generally take a greedy approach to graph coloring. Nodes are ordered by centrality or degree and then colored according to some maximal criteria. For example, under the Dsatur algorithm: the nodes are ordered according to degree, and the node with the largest degree is colored first. Next the node with the largest saturation degree is colored - the saturation degree is the number of different colors a node is adjacent to. Then, the previous step is repeated until all the nodes are colored. [2]

Deterministic algorithms tend to have a very logical flow and structure. There is an ordering and hierarchical mentality, and it is usually evident how far along the coloring algorithm is. For instance, one can look at the time-series plot of the proportion of nodes that are properly colored relative to their neighbors and this will be a monotonically increasing

function. This is where probabilistic methods differ.

Using a typical probabilistic, centralized approach, nodes are randomly assigned colors. And if there is a conflict, nodes are recolored in some iterative, but still random, manner; the amount of randomness varies depending on the algorithm. For example, if there is a conflict, nodes may randomly select an available channel or some nodes may not even switch channels with some probability. The main problem with probabilistic algorithms is that it is often difficult to make guarantees on how many iterations it will take before channels are properly assigned, and it is also often not evident how far along the algorithm is. A time-series plot for probabilistic algorithms more closely resembles fluctuations in the stock market than a monotonically increasing function. Also, it is not at all obvious whether this method should perform better than a deterministic approach. [16]

Some of the advantages of these centralized schemes are that a proper coloring can be achieved in a single pass, the central processing unit is knowledgeable of all the nodes, and it is usually evident when schemes will achieve a proper coloring. The main drawback of these methods, however, is that this central processing unit slows down the channel assignment and is also part of the infrastructure we are trying to get rid of. Distributed coloring algorithms, which we will cover next, remove this requirement for a central processing unit.

3.2 Distributed Coloring

In contrast, distributed algorithms are performed by individual agents making the decisions about which channel to occupy and these individual agents have only local awareness about the channels of their neighbors. One of the main advantages of distributed algorithms is that they are extremely simple to implement, which is advantageous when using low cost micro-controllers. Agents do not need to consider if the entire graph is properly colored as in the centralized case, but merely consider if they are properly colored in regards to their local neighborhood.

Again, we will discuss deterministic and probabilistic algorithms. In typical distributed, deterministic coloring algorithms there is usually an initial localization/symmetry breaking

step. During this step, a leader(s) is(are) chosen or a central point is determined that allows the proceeding algorithm to have the collective behavior necessary to achieve a proper coloring. In [11], sensors localize themselves based on their distance to the center or 'sink' of the sensor network. This information allows the sensors to place themselves into layers/concentric rings for symmetry breaking. Once this occurs, nodes individually select channels based on communication that is passed between neighbors, typically greedy methods. Again, to an observer the progression of the proper coloring is usually clearly visible and it is evident how far along the algorithm is.

Under a distributed, probabilistic approach, there is usually still some initialization process, but the selection of channels has some amount of randomness to it; some algorithms even allow nodes to select conflicting channels. Again, depending on the algorithm, it may not be at all evident how far along it is to a proper coloring. [9] [18]

The main drawbacks of the majority of the approaches we have covered so far is that there is a level of assumed communication between agents or the use of a central processing unit. In the distributed case, nodes are usually able to tell neighbors which channels are available or what their degrees are. And, in the centralized case, the CPU has full knowledge of the communication graph. Unfortunately, this communication layer/knowledge does not exist under our system; if nodes are not properly colored, it is assumed that they cannot pass any information. This forces us to consider a passive approach to channel assignment, one in which agents assign themselves channels but do not assign channels to others.

Chapter 4
MODIFIED ALOHA PROTOCOL

4.1 Description

In order to distributively and passively assign channels to agents utilizing the Beacon system, we propose a modification of the ALOHA protocol (Additive Link On-Line Hawaii system). Under the ALOHA protocol, an agent broadcasts its message at will to a centralized server. If the agent does not receive a confirmation from the server that the message was received, it waits a predetermined period of time before rebroadcasting the message. [21] Under the Beacon system, agents are always broadcasting, but we adapt this protocol so that if agents do not receive a WiFI confirmation from a neighbor, they switch to a new random channel; all channels have an equal probability of being picked, including the channel the agent was previously on. Not receiving a WiFI confirmation means that the neighboring agent they are trying to communicate with cannot determine their VIN due to interference. This channel assignment protocol is similar to '*the trivial algorithm*' analyzed in [12], however, it does not have several of the restrictions associated with it such as having a forbidden color palette; our protocol is also asynchronous. Below is an outline of the events of our algorithm.

Order of events for all agents:

- `scan all channels for signals`
- `if a neighbor is on your channel, switch channels and start over, else, continue`
- `get VINs of neighbors from channels without interference`
- `send a WiFI request to all neighbors the agent has a VIN for`
- `wait one cycle for WiFI confirmation`

- `if any WiFI request is not confirmed, change channels`

Using the agents from $\mathcal{G}_1$ in Figure 2.1 as an example:

- (1) cannot get VINs of (2) and (3) to send WiFI requests to due to interference
- (2) and (3) can get VIN of (1) and each send a WiFI request to (1)
- (2) and (3) do not receive WiFI confirmations due to interference
- (3) picks a new channel at random resulting in $\mathcal{G}_2$
- (2) picks a new random channel but it just happens to be the same channel it was on
- (2) and (3) send a WiFI request to (1)
- (1) sends WiFI confirmations to (2) and (3)
- all nodes in $\mathcal{G}_2$ can communicate

Something to note from this example is that we require agents to switch channels if a neighbor occupies the same channel as them, i.e., the proper coloring of $\mathcal{G}$. Although there are some situations where agents could still communicate without this requirement, we have included it due to the fact that reflections off objects may cause agents to interfere with each other even if they are not trying to communicate with a common neighbor.

We can show that this algorithm leads to a proper coloring, provided one exists. We will use a tree diagram to supplement our proof, and the proof follows after this brief illustration. Without loss of generality, consider $\tilde{\mathcal{G}}_1$ in Figure 2.1. If we start with the coloring shown in the figure, then time step t_1 in Figure 4.1 shows the tree diagram of possible outcomes from this initial assignment. For this scenario we assume that there are only three channels available, red (r), blue (b), and green (g) and that node 2 is the first to run the algorithm followed

by node 3 - during each time step, a single node makes the decision to switch channels or not.[1] We let the vector $\mathbf{c} = (\ r \quad b \quad b\ )^T$ represent the coloring of $\tilde{\mathcal{G}}_1$, where c_i is the color of node i.

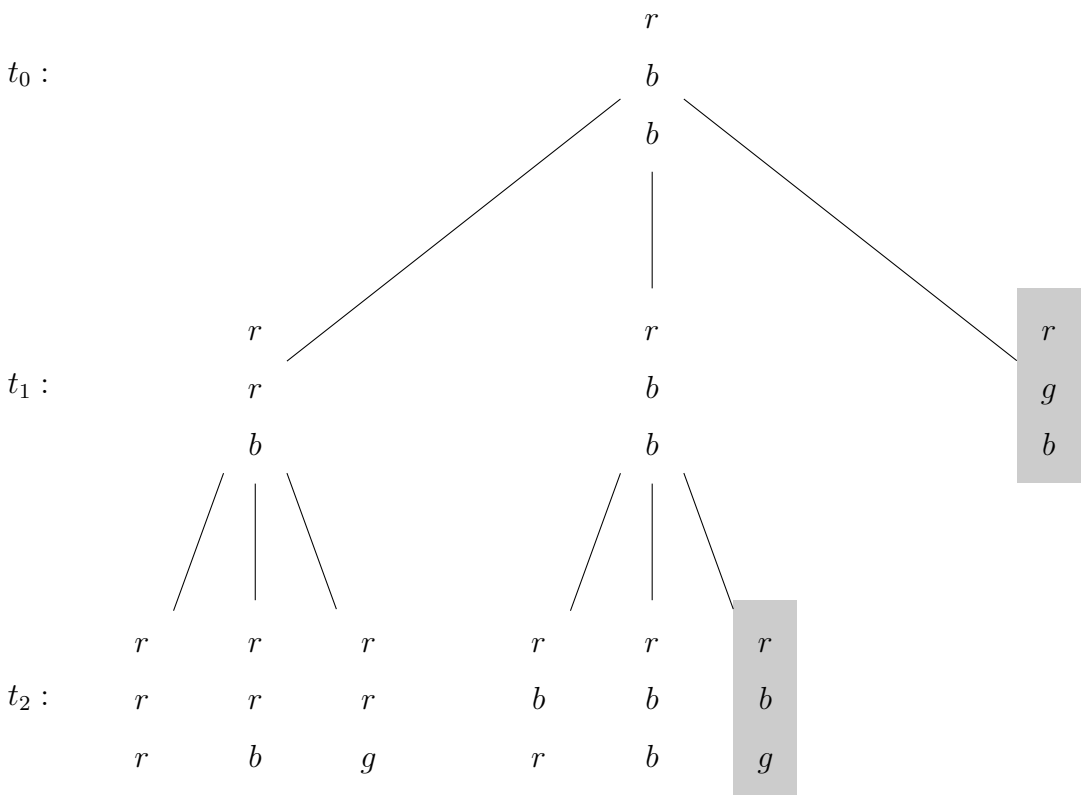

Figure 4.1: Tree diagram depicting possible outcomes of the channel assignment algorithm for two time steps.

Referring to the figure, since node 2 has a conflict[2], it will randomly pick a new channel.

[1]This is an asynchronous coloring algorithm where agents take turns selecting channels as opposed to a synchronous algorithm where all agents select channels at the same time. This method was chosen for several reasons: its progression is easier to follow and it more closely mimics real world robots/sensors whose clock cycles may not be in sync. We can justify this last statement if we assume that the processing of channels and initiating/confirming handshakes takes the same amount of time for each agent. Then, we further assume that the clock cycles of each micro-controller is mutually out of phase with every other agent. With these assumptions, channel selection will proceed in an ordered fashion with the agent with the lowest phase starting first.

[2]We say that a node has a *conflict* if according to the order of events previously explained the node would

The possible outcomes of each of the new selections is given in time step t_1; only one of these outcomes produces a proper coloring. Terminating branches after proper colorings, depicted in the gray boxes in time steps t_1 and t_2, prevents counting unnecessary assignments further down. Branches for alternative outcomes also terminate for nodes if they would not have to switch channels, i.e., if they did not have a channel conflict with their neighbors (not present in this scenario). During time step t_2, node 3 again has three possible outcomes for each of the improperly colored assignments that result from the previous time step. Then, the probability that the graph is properly colored after a certain number of time steps, $t_1, t_2, \ldots, t_f$, is the probability that the graph is properly colored after t_f conditioned on the graph not being properly colored during any previous time steps. For this scenario, the graph is not properly colored at t_0, there is one out of three outcomes in t_1 that leads to a proper coloring. There is one out of six outcomes in t_2 that leads to a proper coloring and the conditional probability that a proper coloring was not achieved in the previous time step is two out of three. Then the probability that the graph is properly colored after time step t_2 is $0 + \frac{1}{3} + \frac{1}{9} = \frac{4}{9}$.

This is not an efficient way to find proper colorings of graphs, but it provides us with all the necessary elements to prove that this algorithm always leads to a proper coloring, provided one exists.

Theorem 1 *If a proper coloring of $\tilde{\mathcal{G}}$ is possible, the modified ALOHA protocol will achieve a proper coloring in the limiting case as $t_f \to \infty$, i.e., $\bar{P} \to 0$, as $t \to \infty$, where $\bar{P}$ is the probability that the graph is not properly colored.*

Proof 1 *Firstly, all possible outcomes according to a node assignment ordering are present in the tree. This is because each branch of the tree contains all permutations of colorings for subsequent nodes. Secondly, the conditional probability that the graph is not properly colored after each time step is less than or equal to one and is the product of preceding probabilities*

---

switch channels, i.e., if the node does not receive a WiFI confirmation from a neighbor that it sent a WiFI request to.

of the graph not being properly colored. For instance, if we say that $\bar{P}_0$ is the probability that the graph is not properly colored after time-step t_0, then the probability that a graph is not properly colored after time-step t_m is $\bar{P}_0 \cdot \bar{P}_1 \cdot \bar{P}_2 \cdots \bar{P}_{m-1} \cdot \bar{P}_m$. In the limiting case, if the graph is colorable, an infinite number[3] of branches will have a proper coloring assignment which also means that there will be an infinite number of probabilities of the graph not being properly colored that are less than one. Therefore, in the limiting case if a proper coloring is possible, the probability of the graph not being properly colored goes to zero.

A variation on this modified ALOHA protocol was investigated where the likelihood of switching channels due to a conflict was dependent on the number of neighbors. We found that the best performing relationship between the number of neighbors and the probability of switching channels was the linear relationship[4]

$$\texttt{prob\_switch}_i = \frac{2k - 3 - |\mathcal{N}(i)|}{2(k-2)} \tag{4.1}$$

where i is the node with a conflict. This function yields a probability of switching equal to one for nodes with a single neighbor and decreases to a probability of switching equal to one half for nodes with the maximum, $k-1$ neighbors, where again, k is the number of available channels.

As it turns out, decreasing the likelihood of switching channels for nodes with many neighbors performed worse than when all nodes had an equal probability of switching due to conflict; originally, we felt that this asymmetric method may perform better since it is less likely that nodes with few neighbors will cause conflicts when changing channels compared to nodes with many neighbors. Furthermore, having some nodes 'refuse' to switch channels more closely resembles the coloring algorithms previously studied and verified to achieve a proper coloring. Figure 4.2 shows the results from one of these comparisons on a random graph with 60 nodes and $\Delta + 3$ channels. The figure shows the time-series analysis of the

[3]This is because the permutations of colorings are repeated every n time steps, where $n = |\mathcal{V}|$ is the number of nodes in the graph.

[4]Other relationships considered were polynomial and exponential.

proportion of nodes that have a channel conflict with a neighbor. For this comparison, a random, connected graph with 60 nodes was generated and initialized with a random channel assignment to each node. The graph and initial channel assignment was stored and used on both simulations. After initialization, the channel assignments for the nodes proceeded according to the modified ALOHA protocol where nodes select channels in turn.[5] It can be seen in the first trial that having a constant probability of switching allowed the graph to achieve a proper coloring in approximately have the number of iterations. Not all simulations had this large of a difference in the number of iterations to a proper coloring, but it appears that switching channels due to a conflict allows for a faster search of the coloring space, regardless of the node's degree. In the next section we will look at the evolution of the conditional probability of not being properly colored over time.

4.2 Probability of Conflict

Due to the probabilistic nature of our channel assignment algorithm, we are concerned with the probabilities of conflict and the likelihood of generating a proper channel assignment as opposed to deterministic measures. First, we will determine what the probability of a conflict after initialization is. Again, it is assumed that during initialization nodes randomly pick channels with each channel having an equal probability of being selected.

Theorem 2 *After initialization, the probability of a conflict in $\tilde{\mathcal{G}}$ for some node i is,*

$$P_i(j \cup l \cup \cdots \cup q) = P_i^{\tilde{\mathcal{N}}(i)} = \sum_{n=1}^{x} \frac{\binom{x}{n}}{k^n}(-1)^{n+1}, \tag{4.2}$$

where $x = |\tilde{\mathcal{N}}(i)|$, the number of neighbors of node i in the graph $\tilde{\mathcal{G}}$.

Proof 2 *Eqn. 4.2 is a specific case of the Inclusion-Exclusion Principle [19] so the proof is straightforward. For a node i with neighbors j, l, $m, \ldots q$, the Inclusion-Exclusion Principle*

[5] The order of channel selection for the nodes was by node number, i.e., $1, 2, 3, \ldots, 60$, but the placement of nodes was completely random.

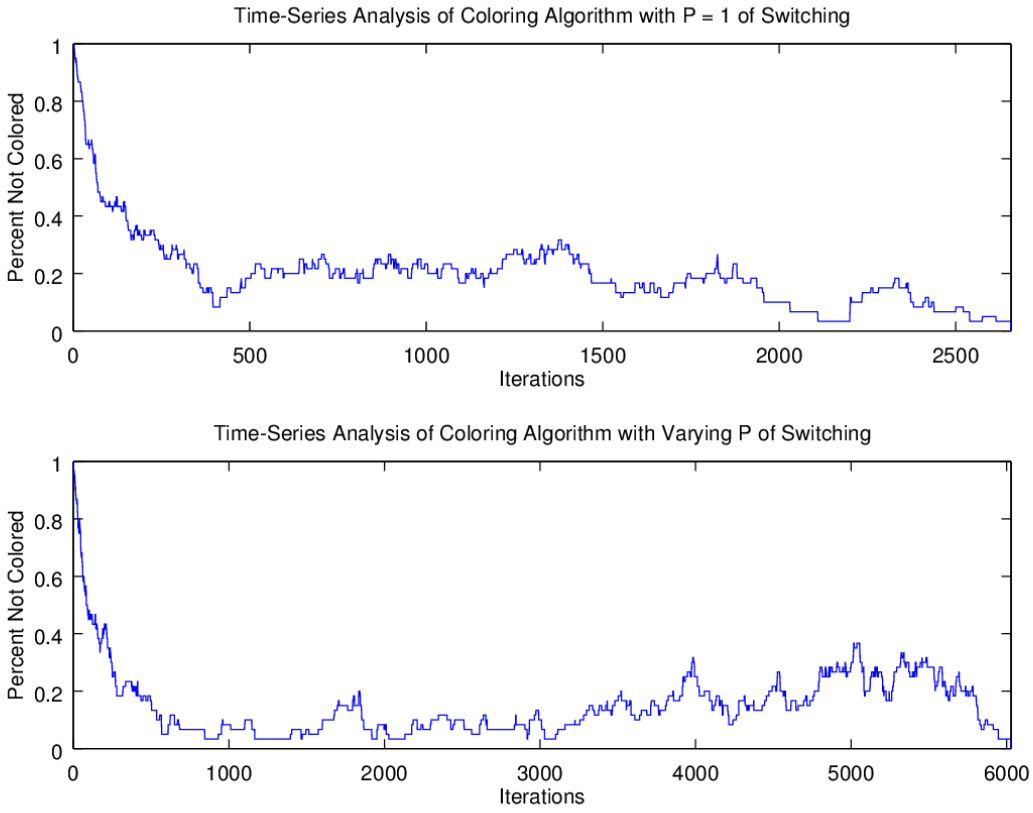

Figure 4.2: A random graph was generated and random channels were assigned to 60 nodes. Then, channels were assigned according to the modified ALOHA protocol.

states that the probability that any of the events $c(i) = c(j),\ c(i) = c(l), \ldots, c(i) = c(q)$ *occurs is,*

$$\begin{aligned} P_i^{\tilde{N}(i)} = {} & P_i(j) + P_i(l) + P_i(m) + \ldots + P_i(q) - P_i(j \cap l) - \\ & P_i(j \cap m) - P_i(l \cap m) - \ldots - P_i(m \cap q) + \\ & P_i(j \cap l \cap m) + \ldots + P_i(l \cap m \cap q) - \\ & P_i(j \cap l \cap m \cap q) - \ldots \end{aligned}$$

This is simply the sum of the probabilities of all the individual events $c(i) = c(j),\ c(i) =$

$c(l), \ldots, c(i) = c(q)$ occurring minus the probabilities of all combinations of pairs of events $c(i) = c(j) = c(l)$, $c(i) = c(l) = c(m), \ldots$, occurring plus the probabilities of all combinations of triplets $c(i) = c(j) = c(l) = c(m), \ldots$ etc., alternating addition and subtraction until the intersection of all events. That is,

$$P_i^{\tilde{\mathcal{N}}(i)} = \sum_{j=1}^{x} P_i(j) - \sum_{j<l}^{x} P_i(j \cap l) + \sum_{j<l<m}^{x} P_i(j \cap l \cap m) - \ldots + (-1)^{x+1} P_i(j \cap l \cap m \cap \ldots \cap q).$$

And, since the probabilities of individual events are independent and equal, i.e., $P_i(j) = P_i(l) = \ldots = 1/k$, the probabilities of individual events, pairs, triplets,... is $1/k$ raised to the powers 1, 2, 3,... Then, the total number of individual events, pairs, triplets,... is the combination $\binom{x}{1}$, $\binom{x}{2}$, $\binom{x}{3}, \ldots$ Therefore, the total probability that any of the events occurs is,

$$P_i^{\tilde{\mathcal{N}}(i)} = \sum_{n=1}^{x} \frac{\binom{x}{n}}{k^n} (-1)^{n+1}.$$

Now that we know what the probability of conflict is for all nodes after initialization, we want to use this to determine the evolution of the probability of the graph not being properly colored. And, because our algorithm is asynchronous and dependent on the ordering of nodes selecting channels, we need to turn to specific examples to gain a better understanding of this evolution.

4.3 Analysis of the Modified ALOHA Protocol via Markov Chains

Initially, we tried to use Markov Chains to study the evolution of channels. It was thought that this method would provide a more efficient way to analyze the evolution of probabilities when compared to the tree diagrams outlined in Section 1. To do this, we created the probability matrix $[P]^i$ for node i as follows:

- $[P]^i_{11}$ is the probability that node i stays on channel 1, given that it is already on this channel. There are two ways this can happen: (1) there is not a conflict in channels so

node i stays on this channel and (2) there is a conflict so node i switches but chooses channel 1 again. This probability is given by

$$[P]^i_{jj} = (1 - P_i^{\tilde{\mathcal{N}}(i)}) + \frac{P_i^{\tilde{\mathcal{N}}(i)}}{k}, \tag{4.3}$$

where j is the channel number, the first term on the right hand side is the probability that node i does not have a conflict, and the term $P_i^{\tilde{\mathcal{N}}(i)}/k$ is due to the fact that even if a node chooses to switch channels, there is a $1/k$ probability that it will stay on the same channel.

- $[P]^i_{12}$ is the probability that node i switches to channel 2, given that it was on channel 1. Unlike $[P]^i_{11}$, there is only one way for this to happen: there is a conflict in channels so node i switches and chooses channel 2. This probability is given by

$$[P]^i_{jl} = \frac{P_i^{\tilde{\mathcal{N}}(i)}}{k}, \tag{4.4}$$

where node i is on channel j and switches to channel l.

- And, $[P]^i_{1k}$ is the probability that node i switches to channel k, given that it was on channel 1. Similarly to $[P]^i_{12}$ there is only one way for this to happen.

Rows $[P]^i_{2j}$, $[P]^i_{3j}, \ldots [P]^i_{kj}$ are found similarly, substituting in the appropriate channels 2, 3,$\ldots k$ for channel 1 in the above outline.

Then, the probability matrix for node i is,

$$[P]^i = \begin{bmatrix} \alpha & \beta & \cdots & \beta \\ \beta & \alpha & \cdots & \beta \\ \vdots & & \ddots & \\ \beta & \beta & \cdots & \alpha \end{bmatrix},$$

where $\alpha = (1 - P_i^{\tilde{\mathcal{N}}(i)}) + P_i^{\tilde{\mathcal{N}}(i)}/k$, and $\beta = P_i^{\tilde{\mathcal{N}}(i)}/k$. Using the graph in Figure 4.3 with 4 available channels as an example, $[P]^1 = [P]^2 = \ldots [P]^4$ is,

$$[P]^1 = \frac{1}{256}\begin{bmatrix} 145 & 37 & 37 & 37 \\ 37 & 145 & 37 & 37 \\ 37 & 37 & 145 & 37 \\ 37 & 37 & 37 & 145 \end{bmatrix}.$$

Therefore, if node 1 starts on a particular channel at t_0, the channel probability vector for node 1 will be $p(t_0) = [1\ 0\ 0\ 0]^T$, where the elements of the probability vector are for the channels 1, 2, 3, 4. Then, after 1 time step the probability vector will be,

$$p(t_1) = [P]^1 p(t_0) = \frac{1}{256}[145\ 37\ 37\ 37]^T.$$

Unfortunately, as can be seen from the example, this probability matrix only serves to even out the probabilities of nodes being on any specific channel. We would need to assign specific channels to the nodes in order to determine how the channel assignment would progress. However, this would bring us right back to the analysis via tree diagrams. Therefore, we require other means of evaluating the channel assignment algorithm.

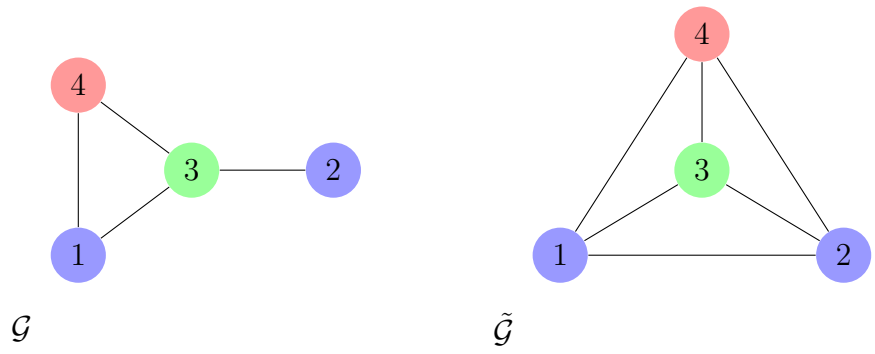

Figure 4.3: Network of robotic agents. This network would require at least 4 channels to be conflict free, since $\tilde{\Delta} = 3$.

4.4 Channel Assignment Under Brownian Motion

In order to determine the effectiveness of the channel assignment protocol without the aid of analytic methods, nodes were simulated moving randomly in a confined space using Processing visualization software; one frame of these simulations is shown in Figure 4.4. This movement allowed countless graphs to be tested and the communication success of the agents to be monitored throughout. Some things to note about these simulations are:

- Again, edges between nodes indicate that nodes are within sensing distance of each other but not necessarily that they can communicate.
- All nodes move with a constant speed and collisions with walls and other nodes are completely elastic.
- The percentage of time steps that nodes were able to successfully communicate with all neighbors within their sensing range was monitored and recorded.[6]
- The simulations were allowed to run for 150,000 time steps(approximately 45 minutes).[7]

The evolution of the communication success percentages for one of these simulations can be seen in Figure 4.5. From the figure, it can be seen that the average communication success for this simulation was centered just above forty percent. It should be emphasized that the low communication success for this simulation is merely the communication success with *all* neighbors throughout the simulation and does not reflect the percentage of time steps that agents could not communicate with individual neighbors. There are large portions of the simulation when agents have more neighbors than channels and can not possibly be properly colored, but they can still communicate with certain neighbors that they do not have a conflict with.

[6]The communication success percentage for agents can be seen in the bottom banner of Figure 4.4.

[7]This was the amount of time steps needed for the worst performing simulations to reach a reasonable steady state of communication success.

These simulations were iterated while individually changing the number of available channels, number of nodes, the node's sensing range, and the simulation area. Figures 4.6-4.9 show the final communication success percentages for all agents for these simulations. And, they can be used when making cost/benefit decisions while utilizing the modified ALOHA protocol, such as how many channels are needed to provide sufficient communication success or the optimal number of agents that should be used to occupy a certain space given a specific number of channels.

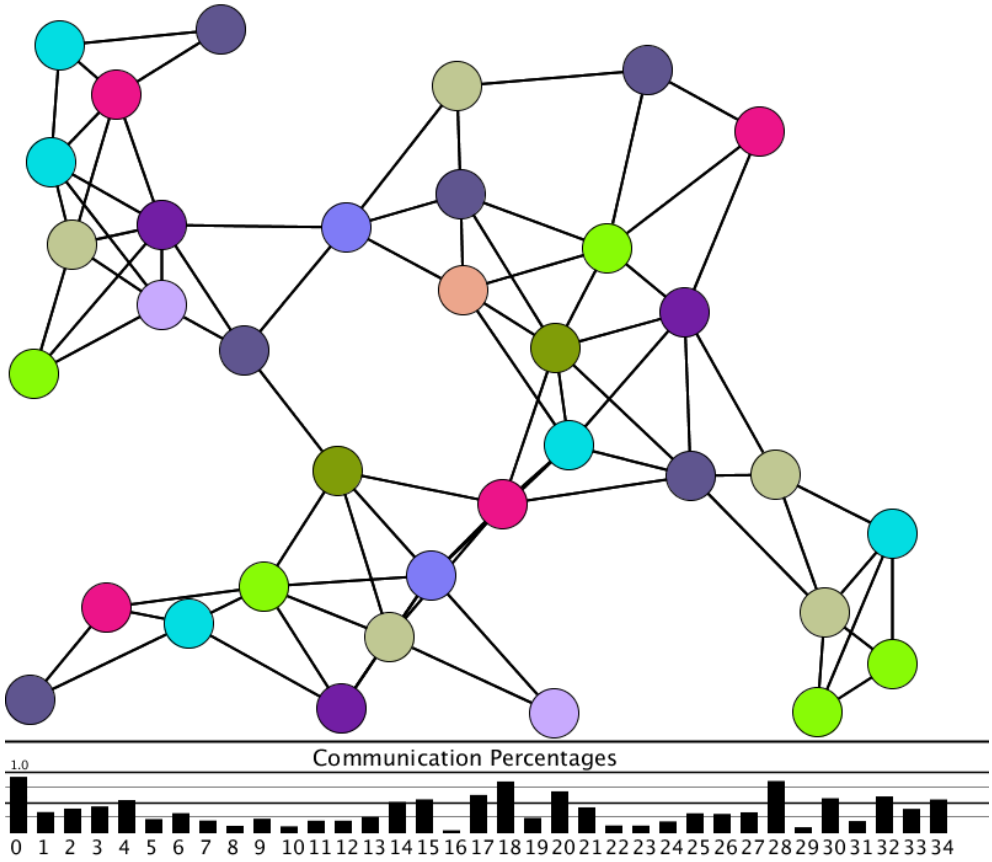

Figure 4.4: Single frame of simulation of agents utilizing the modified ALOHA protocol for channel assignment under Brownian motion.

As previously stated, the benefit of observing the performance of the algorithm under this motion is that a countless number of configurations/graphs are tested with their average

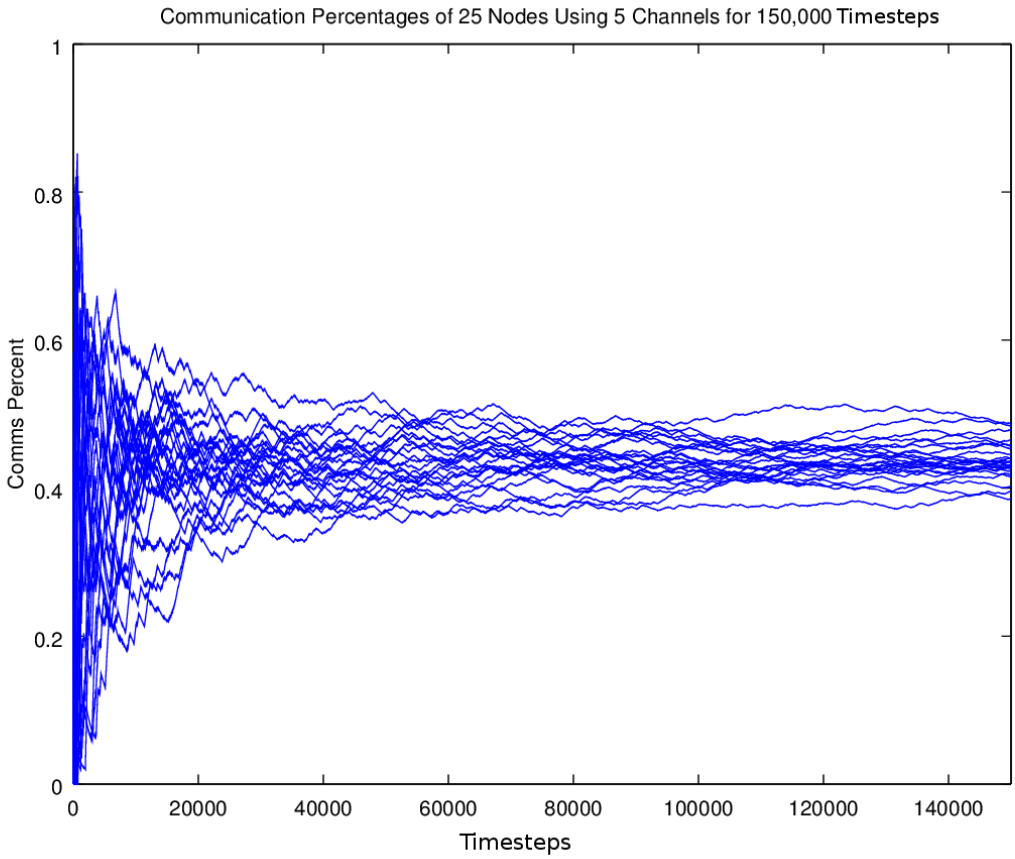

Figure 4.5: Evolution of communication success percentages of 25 nodes utilizing the modified ALOHA protocol for channel assignment with 5 available channels under Brownian Motion.

communication success providing a measure of performance. But, due to this averaging, it is not clear how well the algorithm performs under specific circumstances. If it is desired, certain graphs with a physical significance to robot dynamics or wireless networks can always be studied separately. In the next section, we will look at the performance of this protocol during consensus dynamics.

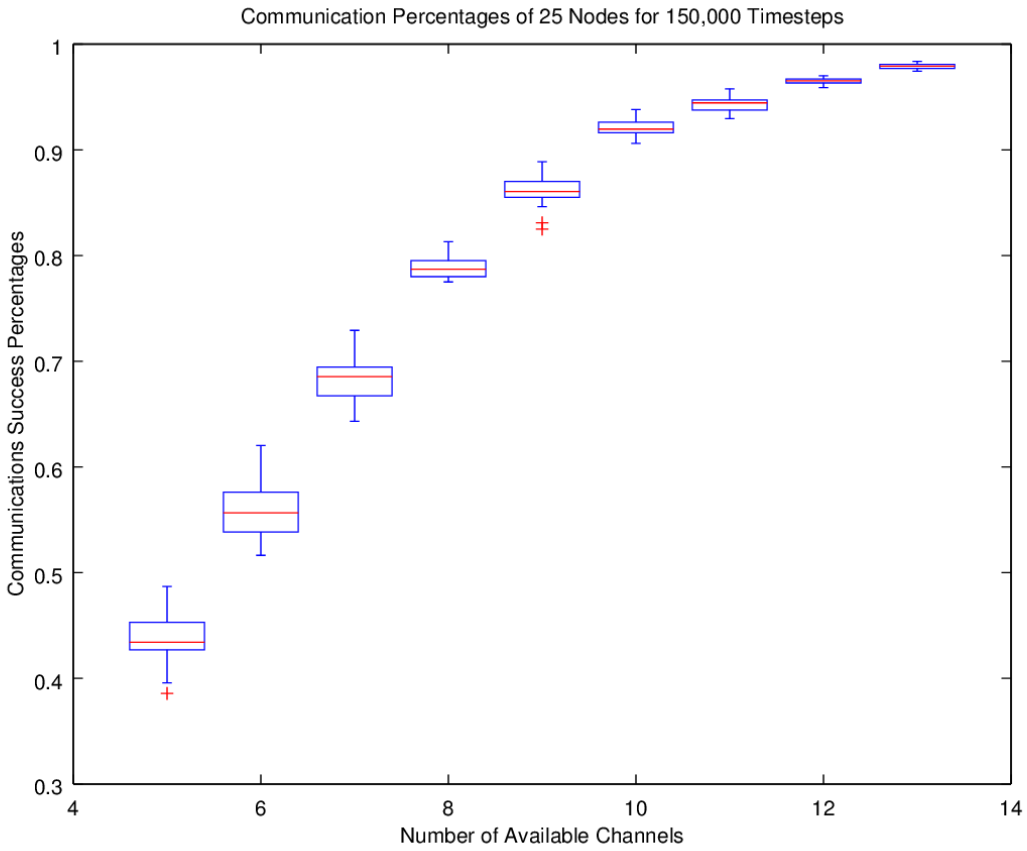

Figure 4.6: Statistical analysis showing mean (red bar), quartiles, and outliers (red crosses) of final communication success percentage values after 150,000 time steps for various numbers of available channels.

4.5 Modified ALOHA Protocol for Sensor Networks

Before we investigate how the modified ALOHA protocol affects the dynamics of robotic agents, we want to determine how this protocol affects the consensus dynamics that drive the movements of these robots. To do this, we looked at how channel assignment affects the consensus for measurements taken by individual sensors in a static sensor network. For this analysis, random connected graphs were generated in Processing and uniform, random initial readings were assigned to each node/sensor between 1 and 100. Then, the sensors ran

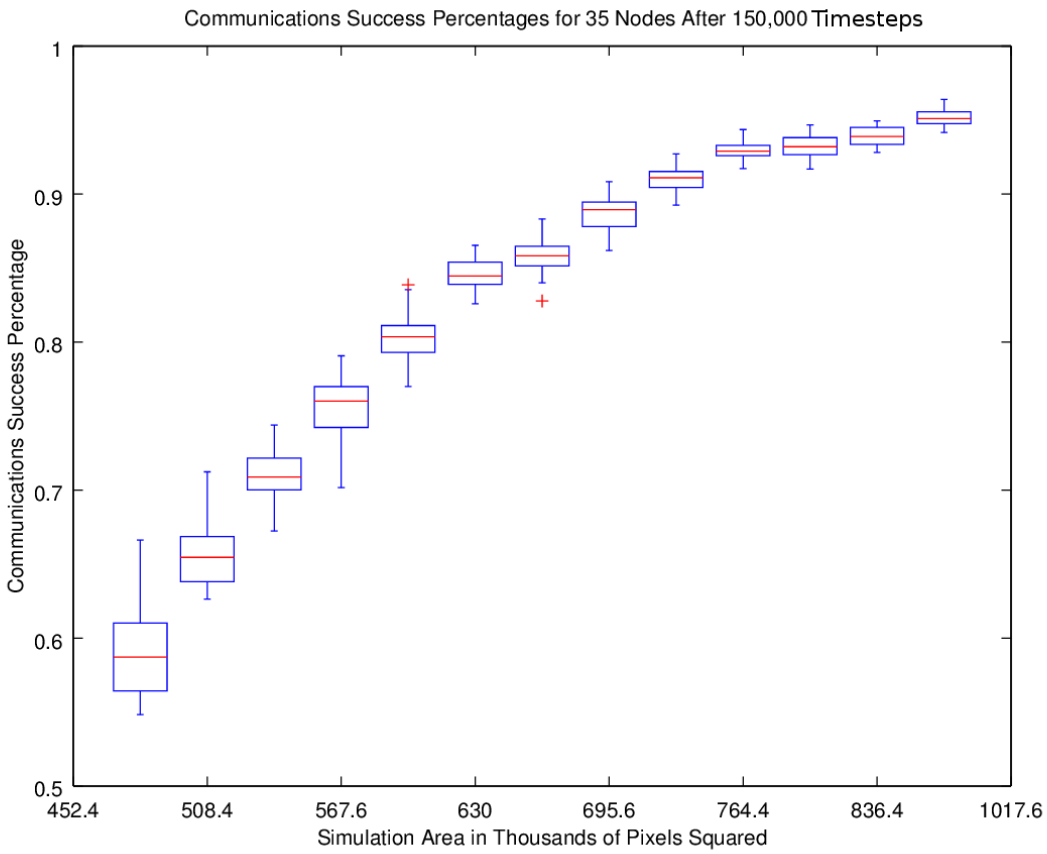

Figure 4.7: Statistical analysis showing mean (red bar), quartiles, and outliers (red crosses) of final communication success percentage values after 150,000 time steps for various numbers of available simulation area.

consensus dynamics according to the two schemes shown in (4.5) and (4.6). Here the sensor measurement of i is x_i and $\breve{\mathcal{N}}(i)$ is the subset of neighbors of i that can be communicated with. The first of these schemes, (4.5), is the consensus dynamics for an undirected graph and the second, (4.6), is the consensus dynamics for an undirected graph with the added communication constraints imposed by the modified ALOHA protocol. And, although both schemes are for undirected graphs, the communication constraints imposed by our protocol have the effect of creating a directed graph. Figure 4.10 shows a single frame of one of these simulations.

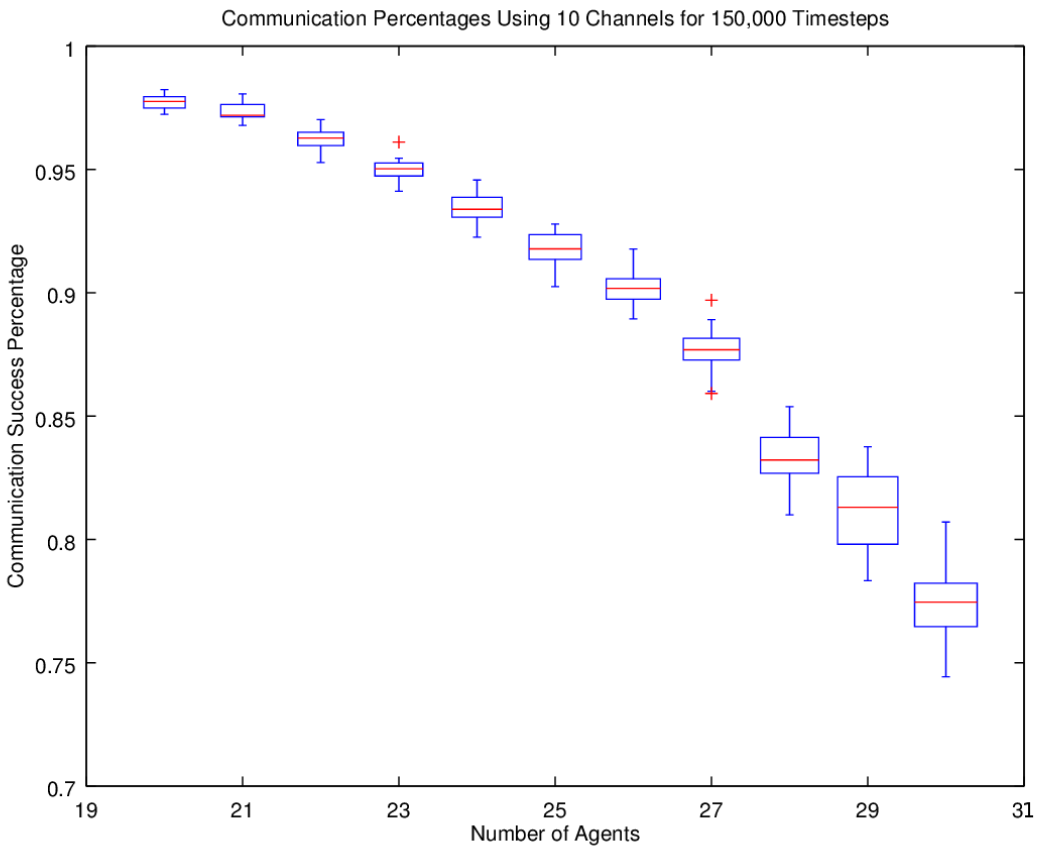

Figure 4.8: Statistical analysis showing mean (red bar), quartiles, and outliers (red crosses) of final communication success percentage values after 150,000 time steps for various numbers of agents.

$$\dot{x}_i = \sum_{j \in \mathcal{N}(i)} (x_j - x_i) \tag{4.5}$$

$$\dot{x}_i = \sum_{j \in \check{\mathcal{N}}(i)} (x_j - x_i) \tag{4.6}$$

For an undirected, connected graph running consensus dynamics, the sensors converge to the average of the initial readings of the sensors, i.e., [15],

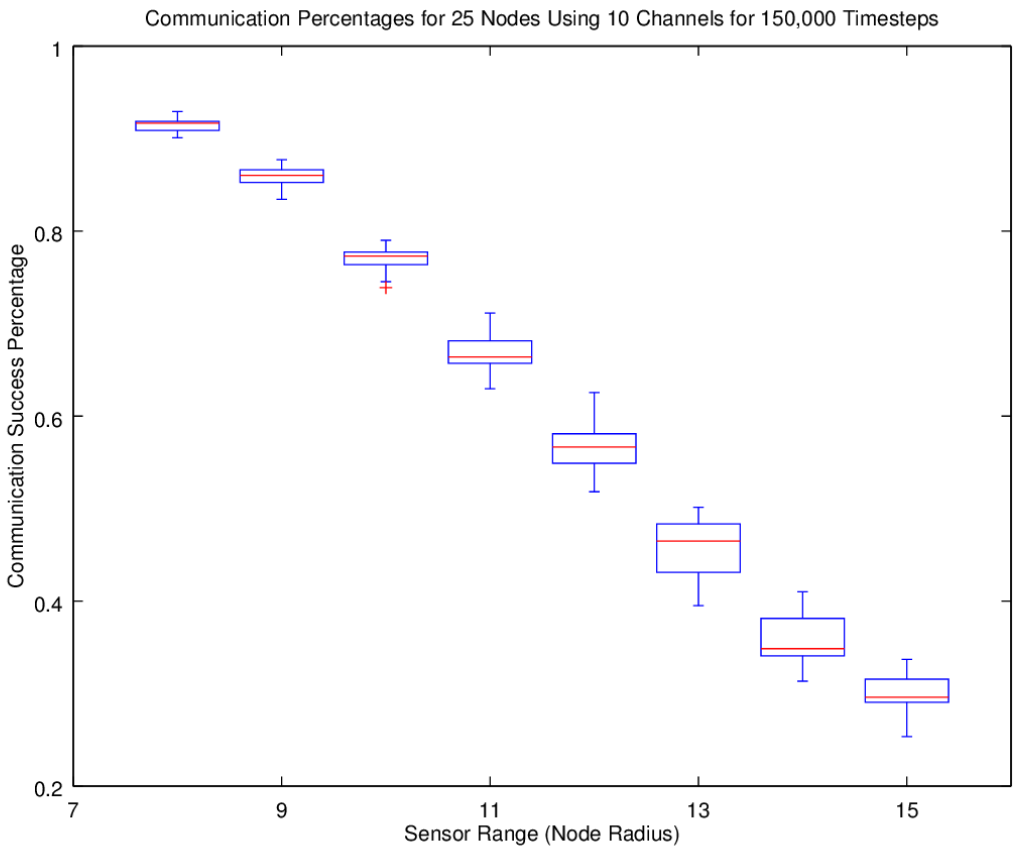

Figure 4.9: Statistical analysis showing mean (red bar), quartiles, and outliers (red crosses) of final communication success percentage values after 150,000 time steps for varying sensor ranges.

$$x_f = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} x_0(i) \tag{4.7}$$

where x_f is the final value for consensus and $x_0(i)$ is the initial reading of node i. This was verified and can be seen in Figure 4.10 where the initial average sensor reading and final consensus value under consensus dynamics agree. However, the figure also shows that the final consensus value using the modified ALOHA protocol does not agree with the initial average. This is due to some nodes not being able to properly communicate with neighbors

throughout the consensus dynamics using the modified ALOHA protocol. This causes the sensor average to change with the flow of information according to the sensors' channel assignments. As previously stated, due to this constraint on the flow of information, the graph becomes directed as some nodes are prevented from passing information to neighbors. This is an important aspect to consider when using this protocol because it leads to biasing of the information passed between nodes. Unfortunately, without conducting case by case analyses, it is difficult to determine what this bias will be due to the probabilistic nature of the channel assignment protocol. But despite this bias, we can still investigate the effects of the protocol on multi-agent robot dynamics.

4.6 Modified ALOHA Protocol and Control

The Robotics Aerospace and Information Networks lab has a swarm of unicycle robots for conducting real world simulations, and this is the model that was chosen for testing the effects of the modified ALOHA protocol on robot dynamics. Figure 4.11 shows a snapshot of one of these simulations using Processing. During this simulation, agents are represented as triangles with the long side indicating the agent's heading. These agents move according to the bearing consensus dynamics for unicycles given in (4.8)-(4.10)

$$\dot{x}_i = v\cos(\theta_i) \tag{4.8}$$

$$\dot{y}_i = v\sin(\theta_i) \tag{4.9}$$

$$\dot{\theta}_i = \omega + \frac{\gamma}{|\mathcal{V}|}\sum_{j\in\breve{\mathcal{N}}(i)}\sin(\theta_j - \theta_i), \tag{4.10}$$

where again $j \in \breve{\mathcal{N}}(i) \subseteq \mathcal{N}(i)$, and ω and β are parameters that are chosen to produce the desired behavior. [15] As to be expected, the agents that can properly communicate with each other reach bearing consensus, but what is interesting about these dynamics is that, as long as all agents can intermittently communicate with a neighbor, this is enough for the

entire formation to reach bearing consensus.[8] This is also shown in [15], where agents will reach consensus provided that the periodic union of subgraphs is connected.

To adjust these dynamics to produce more desirable results, we can add the simple collision avoidance scheme,

$$v_i = (1 - e^{-|\mathbf{r}_i - \mathbf{r}_j|/D})v, \tag{4.11}$$

where $|\mathbf{r}_i - \mathbf{r}_j| = \min_{n \in \mathcal{C}(i)} \{|\mathbf{r}_i - \mathbf{r}_n|\}$, $\mathcal{C}(i)$ is the set of agents within a cone in front of agent i, and D is a desired separation parameter. This collision avoidance scheme causes agents to slow down exponentially as neighboring agents that are within a cone in front of the agent get closer. The results of this collision avoidance scheme can be seen in Figure 4.12. When simultaneously running bearing consensus and the proposed collision avoidance scheme, the agents generate a coverage formation.

Increasing γ to a sufficiently large value allows this type of formation to be controlled by a rebel agent(s) and several of these tests were conducted. The rebel agent(s) can steer the formation from anywhere as long as they can communicate with at least one node and γ is sufficiently high, but we found that the rebel agent(s) had the greatest control when they were towards the front of the formation. It should be noted that large differences in bearing between the rebel agent(s) and the main formation can lead to splitting the formation, however.

[8] The agents reach consensus with an associated bias described in the previous section.

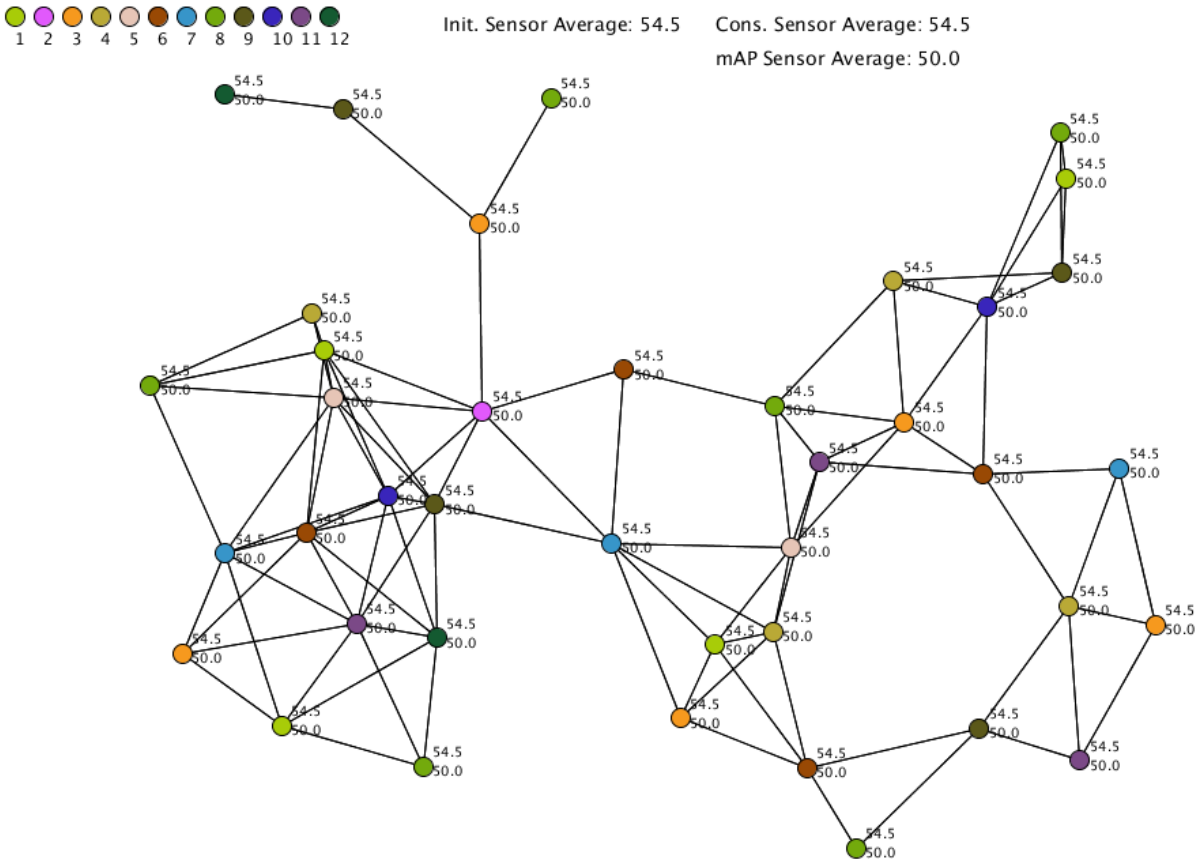

Figure 4.10: Measurement consensus for sensor networks. There are two measurement values associated with each node, one for each consensus protocol used. The top measurement value is determined according to consensus dynamics for a connected, undirected graph. The lower measurement value is determined according to consensus among sensors that can properly communicate with each other using the modified ALOHA protocol. The list of channels are shown for reference in the top left corner and the averages over all sensors from the Initial sensor readings, Consensus sensor readings, and modified ALOHA protocol sensor readings are shown at the top.

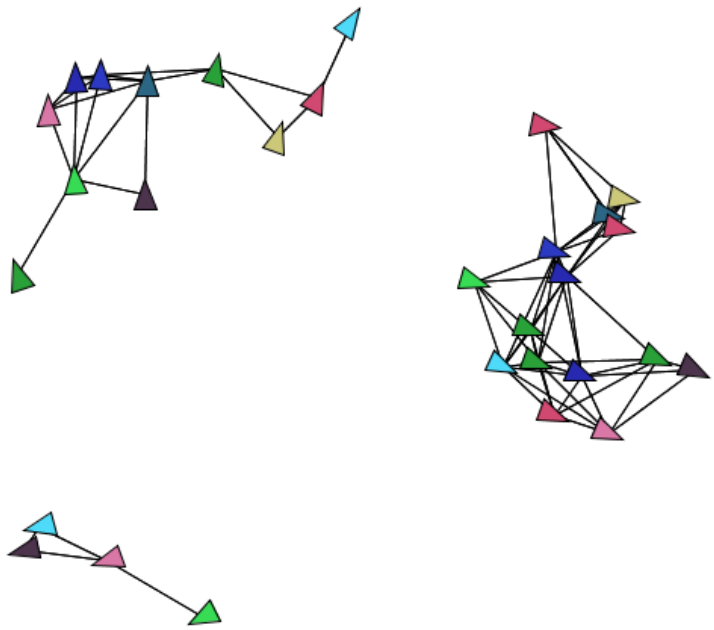

Figure 4.11: Bearing consensus for a unicycle model. Only nodes that can properly communicate with their neighbors share bearing information with adjacent nodes.

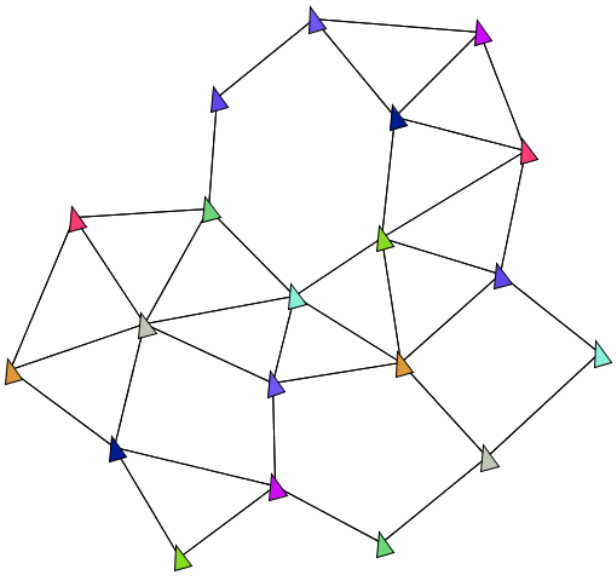

Figure 4.12: Bearing consensus for unicycle models with collision avoidance. Again, only agents that can properly communicate with each other share bearing information, but collision avoidance is independent of communication.

Chapter 5

SUMMARY

5.1 Concluding Remarks

In this thesis, we have discussed our proposal for a distributed, channel assignment algorithm for robotic agents. This protocol is an adaptation of the ALOHA MAC protocol and we have proven that in the limiting case, this protocol converges to a proper channel assignment, provided one exists. We have also found a useful measure for evaluating the success of this protocol under random motions and studied its effects on the flow of information in a graph and how it introduces information bias. Moreover, we have shown that, despite this biasing, we can still achieve the desired behavior for robot dynamics.

It is important to note that this protocol was the solution to a specific problem and is not necessarily suited for all distributed, channel assignment applications. Due to the inability of agents to pass mediating information between neighbors using our proposed local positioning and communication system and the limited capabilities of micro-controllers, we were seeking a passive, channel assignment algorithm that was simple to execute. Channel assignment is an extensively studied field and considerably more advanced methods exist, but this protocol may offer some advantages when cost or processing power are a factor.

5.2 Directions for Further Research

Ultimately, we would like to be able to say more about the convergence of this protocol and would also like to be able to say more about the biasing that occurs. A considerable effort was placed on the former and as shown in this thesis various methods were used to try to quantify this. One further avenue that comes to mind that was not studied here is the similarity of this network to the *threshold protocol* discussed in [15]; instead of the traditional

two states discussed in the text, there are as many states as channels. As for the latter, it was discussed how the probabilistic nature of the algorithm makes it difficult to determine how the protocol affects information flow, but looking at this flow at the time step level may provide answers to the long term behavior.

Finally, we would like to say a little bit about the current progress of the Beacon system. To do this we break up the system into the four components: hardware, signal processing, channel assignment, and localization, and address the three areas that were not covered in this thesis.

- Hardware: We have designed the circuit required for the Beacon system and have an IR LED array prototype, however, we still need to include the IR sensors into the prototype and perform the necessary calibrations to determine the signal amplitude as a function of distance and frequency.

- Signal Processing: We have created the necessary signal processing algorithm for this system, but it has yet to be implemented on the lab's microcontrollers.

- Localization: We are currently working on a Kalman filter for the sensor array that will allow an agent to determine the location of neighboring robots. A scheme that will allow agents to determine their location based on the broadcasted locations of neighboring agents still needs to be developed; a place to start in this area is provided in [20].

If the reader would like access to any code that was used for the simulations in this thesis they can contact me at `vasil.rj@gmail.com`.

BIBLIOGRAPHY

[1] L. Barenboim and M. Elkin. *Synthesis Lectures on Distributed Computing Theory*. Morgan and Claypool, Distributed Graph Coloring: Fundamentals and Recent Developments edition, 2013.

[2] D. Brelaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.

[3] R. E. Cagley, S. A. McNally, and M. R. Wiatt. Dynamic channel spectrum allocation for dynamic use in wireless sensor networks. *Military Communications Conference, IEEE*, pages 1–5, 2006.

[4] A. Chatterjee, S. Deb, and V. Srinivasan. Multiband wireless mesh networks: Frequency aware spectrum allocation and cross-layer optimization. *Fourth International Conference on Communications Systems and Networks*, pages 1–9, 2012.

[5] J. C. Culberson. Iterated greedy graph coloring and the difficulty landscape. *University of Alberta Technical Report*, 1992.

[6] M. Dyer, L. A. Goldberg, and M. Jerrum. Counting and sampling h-colourings. *Lecture notes in computer science*, 2483:51–67, 2002.

[7] M. R. Garey and D. S. Johnson. The complexity of near-optimal graph coloring. *Journal of the Association for Computing Machinery*, 23(1):43–49, 1976.

[8] C. Gavoille, R. Klasing, A. Kosowski, Ł. Kuszner, and A. Navarra. On the complexity of distributed graph coloring with local minimality constraints. *NETWORKS*, pages 12–19, 2009.

[9] J. Hansen, M. Kubale, L. Kuszner, and A. Nadolski. Distributed largest-first algorithm for graph coloring. *Lecture Notes in Computer Science*, 3149:804–811, 2004.

[10] S. H. Hosseini, B. Litow, M. Malkawi, J. McPherson, and K. Vairavan. Analysis of a graph coloring based distributed load balancing algorithm. *Journal of Parallel and Distributed Computing*, 10:160–166, 1990.

[11] I. Jemili, D. Ghrab, B. Derbel A. Belghith, and A. Dhraief. Collision aware coloring algorithm for wireless sensor networks. *9th International Wireless Communications and Mobile Computing Conference*, pages 1546–1553, 2013.

[12] Ö. Johansson. Simple distributed $\delta + 1$-coloring of graphs. *Information Processing Letters*, 70:229–232, 1999.

[13] S. Kornienko, O. Kornienko, and P. Levi. Minimalistic approach towards communication and perception in microrobotic swarms. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2228–2234, 2005.

[14] I. Méndez-Díaz and P. Zabala. A cutting plane algorithm for graph coloring. *Discrete Applied Mathematics*, 156:159–179, 2008.

[15] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, Princeton NJ, 2010.

[16] M. Molloy and B. Read. *Graph Colouring and the Probabilistic Method*. Springer, Berlin, 2002.

[17] R. Muñoz, R. Casellas, and R. Martínez. Dynamic distributed spectrum allocation in gmpls-controlled elastic optical networks. *37th European Conference and Exhibition on Communication*, pages 1–3, 2011.

[18] E. Shamir and E. Upfal. Sequential and distributed graph coloring algorithms with performance analysis in random graph spaces. *Journ. of Algorithms*, 5:488–501, 1984.

[19] A. Slomson. *An Introduction to Combinatorics*. Chapman and Hall, London, 1991.

[20] V. Ugrinovskii. Conditions for detectability in distributed consensus-based observer networks. *IEEE Transactions on Automatic Control*, 58(10):2659–2664, 2013.

[21] J. Zheng and A. Jamalipour. *Wireless Sensor Networks: A Networking Perspective*. IEEE Press, Piscataway NJ, 2009.