CHALMERS

Validating relocation analysis of self-stabilizing MAC algorithms for Mobile Ad-Hoc Networks

Master of Science Thesis in the programme Computer Science

NING HE

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, May 2010

Validating relocation analysis of self-stabilizing MAC algorithms for Mobile Ad-Hoc Networks

Ning He

Examiner: Marina Papatriantafilou

Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden May 2010

Abstract

In this dissertation, I generalize the relocation analysis of MAC algorithms for Mobile Ad-hoc Network by creating different mobility models. The models of Constrained-jump mobility, Parallel-motion and Flock mobility were implemented in TOSSIM. The Constrained-jump model is used for simulating two versions of a MAC Algorithm [3, 9] and comparing their performance. Two functions are derived from the numerical analysis of the Constrained-jump model. These functions are used to predict the throughput of the MAC algorithms in the more complex mobility scenarios: Parallel-motion and Flock mobility. One function considers two parameters (the maximum relocation ratio and the relocation rate); the other one considers one parameter (the similarity ratio). All these parameters reflect the important mobility aspects of the studied models. After the comparison, it turns out that the MAC algorithm's throughput in the Parallel-motion model can be better estimated using the second function. And both functions achieve a good performance predicting the throughput in the Flock model.

Key words: MANET, **mobility model**, **relocation rate**, **similarity ratio**

Acknowledgements

I would like to thank my supervisor Elad Michael Schiller for the suggestion of the problem and guiding to solving it. Thanks also to my examiner Marina Papatriantafilou and my friend Gongxi Zhu for their support. Personal thanks to my family and my friends who have stood by me throughout my education.

Contents

1. INTRODUCTION 6

1.1 Background 6
1.2 Related works 6
1.2.1 Existing Mobility Models 6
1.2.2 Existing MAC protocols 8
1.3 Problem definition 11
1.4 Document organization 12

2 STUDIED ALGORITHMS 13

2.1 Distributed Coloring Algorithm 13
2.2 Improved Distributed Coloring Algorithm 14

3 MOBILITY MODELS AND ANALYSIS 16

3.1 Platform and tools 16
3.2 Terminology 17
3.3 Constrained-jump model 17
3.4 Expectation functions based on Constrained-jump model 21
3.4.1 The motivation 21
3.4.1 Expectation function based on relocation rate and maximum relocation ratio 22
3.4.2 Expectation function based on similarity ratio 23
3.5 Parallel-motion model 23
3.6 Flock model 27

4 CONCLUSIONS 35

5 REFERENCES 36

1. INTRODUCTION

1.1 Background

Mobile Ad-hoc Network (MANET) is a self-configuring network of mobile nodes which communicate with each other by a peer-to-peer topology. MANET has been widely used in our life such as personal mobile equipments and vehicles. It can also be used in the wild to study the behavior of animals or collect useful data from a harsh environment.

Since nodes are mobile in MANET, the network topology may change rapidly and unpredictably over time. Such mobility will seriously affect the network's performance since nodes send and receive data with the same communication channel shared by its neighbors. These changes of the topology will disorder already-established communication and cause collisions in the channel.

Many researchers are dedicating on finding proper MAC algorithms to improve the throughput of MANET. Many algorithms are proposed such as Aloha, Slotted Aloha and FAMA etc. All these techniques aim at increase the throughput and decrease the stabilization time of the network. The current MAC protocol for MANET IEEE802.11 mainly relies on two techniques to combat for the shared radio channel: physical carrier sensing and RTS/CTS handshake strategy.

1.2 Related works

1.2.1 Existing Mobility Models

A mobility model depicts a certain movement scenario in the real life. It describes the movement of nodes, such as their directions, their speeds, etc. Usually, in order to simulate a certain movement of nodes in the real life, we need to create the corresponding model to depict its characters.

Random Walk model

The Random Walk model depicts the simplest movement in the real life. It gives a node four directions to choose at each step. As Figure 1 shows, in each step, each node would randomly choose a direction and move to that direction with a certain speed.

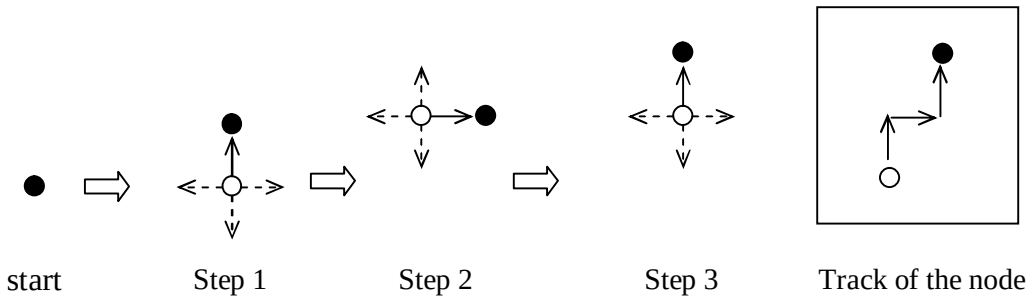


Figure 1. Random Walk model

This model depicts the character of random movement well. However, in this model, a node intends to stay close to its original position due to the little possibility of moving to the same direction in each step. And the real world's situation is much more complicate than just random walk.

Random Waypoint model

The Random Waypoint model is more realistic when depicting the real life's mobility comparing to the Random Walk model. It depicts a group of nodes' movement in a certain area. In Random Waypoint model, nodes are randomly distributed in the simulation area in the beginning, as Figure 2 shows.

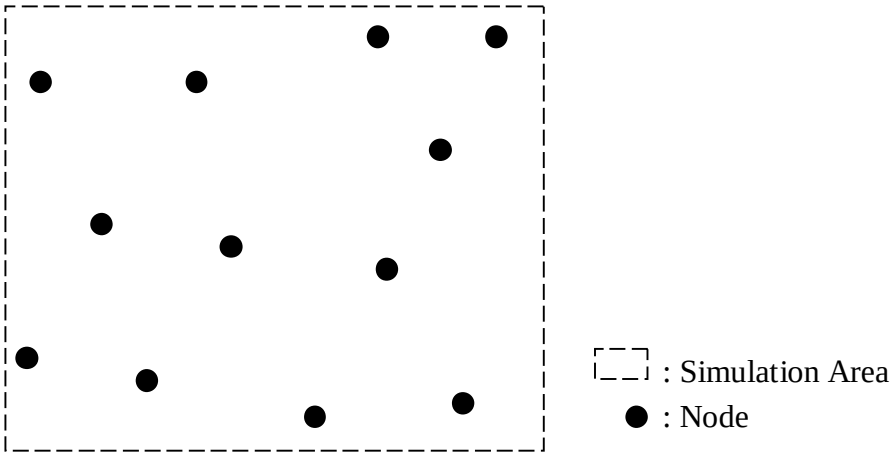


Figure 2. The distribution of nodes in Random Waypoint model

At certain time, each node will randomly choose a target and move to that target with a random speed (Figure 3). After reaching its target, the node will pause for a certain time and start moving again.

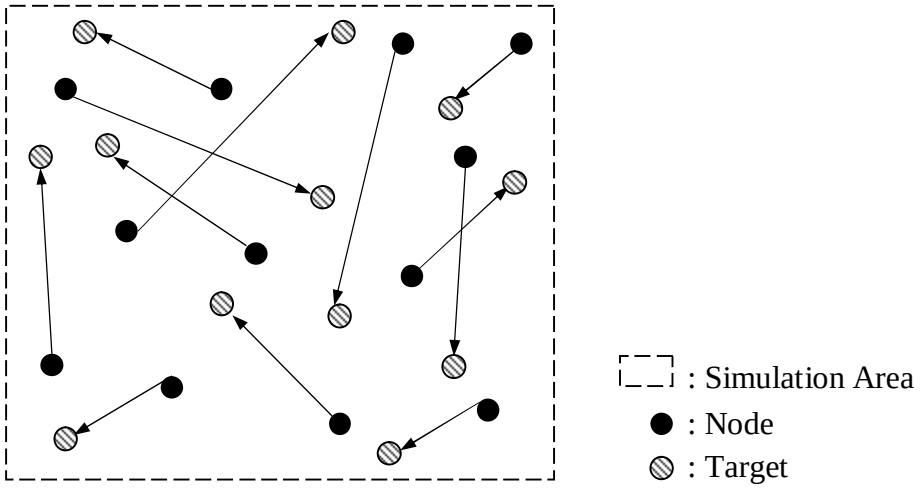


Figure 3. Nodes' movements in Random Waypoint model

Random Waypoint model is more complex than Random Walk model and depicts the real world's mobility better. What it does not consider about is that in the real world, nodes usually stay not far from each other and their movements usually follow certain rules. For example, as students, we usually stay at school with other students. Cars can only run on the road following traffic rules. In the wild, a lot of animals usually stay close to their own species. The reason that we need to consider these situations is because they are typical utilizations when we implement mobile ad-hoc networks in the real world such as using personal digital equipments, monitoring traffic conditions and studying the flock behavior.

1.2.2 Existing MAC protocols

Medium Access Control (MAC) protocol is responsible for coordinating access to the shared radio channel and minimizing conflicts. It plays an important role in wireless communication system.

ALOHA protocol

ALOHA protocol is a simple communication scheme in which each node in a network sends data whenever there is a packet to send.

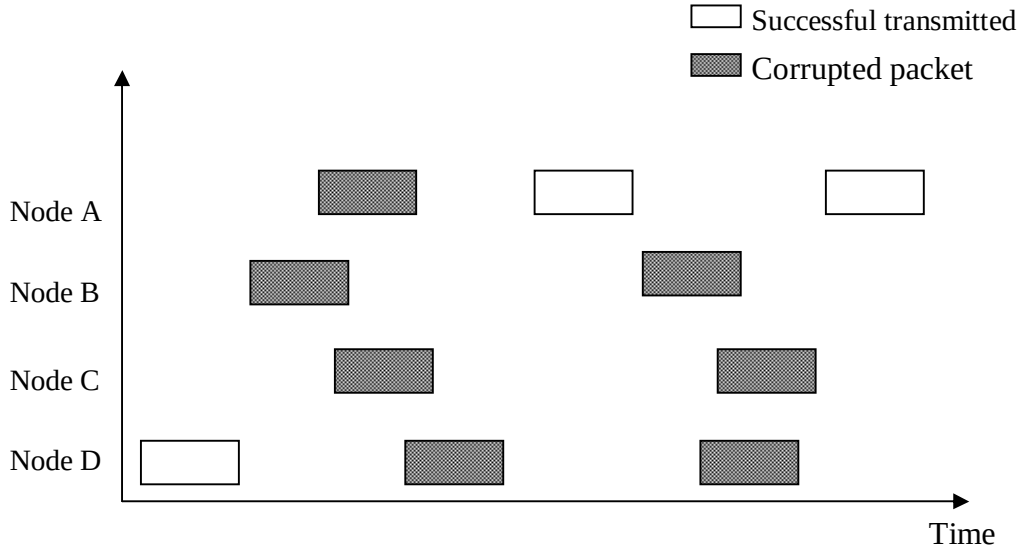


Figure 4. ALOHA protocol

Once the packet successfully reaches the destination (receiver), the next packet will be sent. Figure 4 presents how this protocol works during the communication period. If the packet fails to be received at the destination, it will be sent again. (Figure 5)

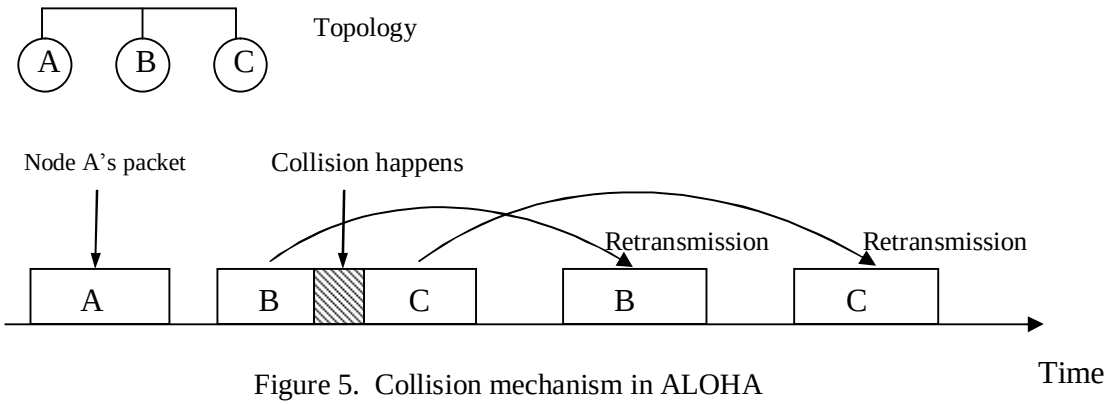


Figure 5. Collision mechanism in ALOHA

The maximum throughput of ALOHA has been proven to be 18.4%, which means that it does not work well in simple wireless broadcast systems. The heavier the communication volume is, the worse the collision becomes. The result is degradation of system efficiency, because when two packets collide, the data contained in both packets is lost and it will cost extra time to send it again.

Slotted ALOHA protocol

By adding some modification, Slotted ALOHA improves the performance of ALOHA protocol with an analytical maximum throughput of 36.8% [8]. As Figure 6 illustrates, the communication channel is divided to several slots whose length is the time necessary to transmit one packet. A node who wants to send a packet will choose an empty slot.

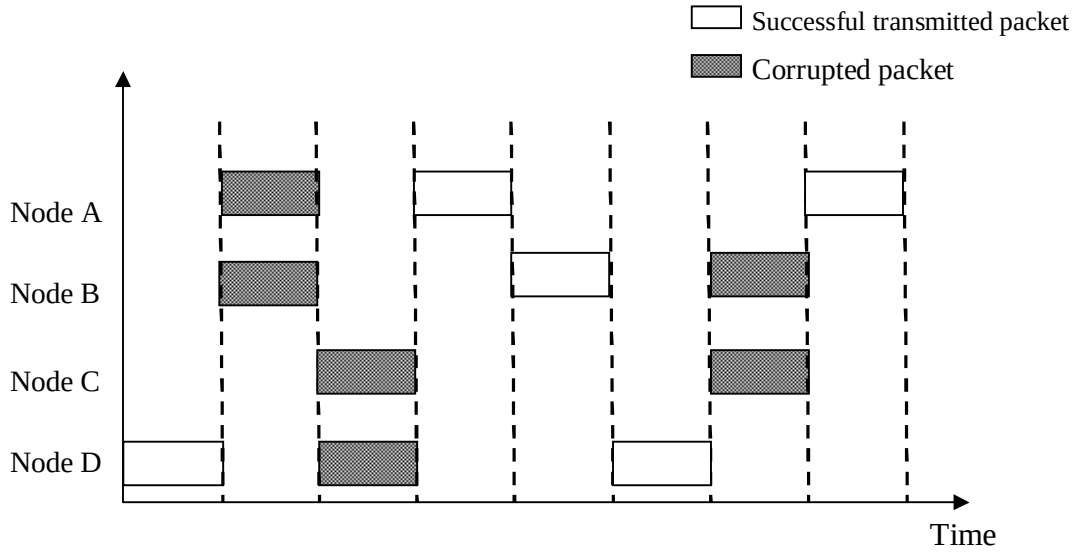


Figure 6. Slotted Aloha protocol

Once the collision happens, each node involves in the collision retransmits at some random time slot. In this way, it reduces the possibility of collision again (As Figure 7 shows).

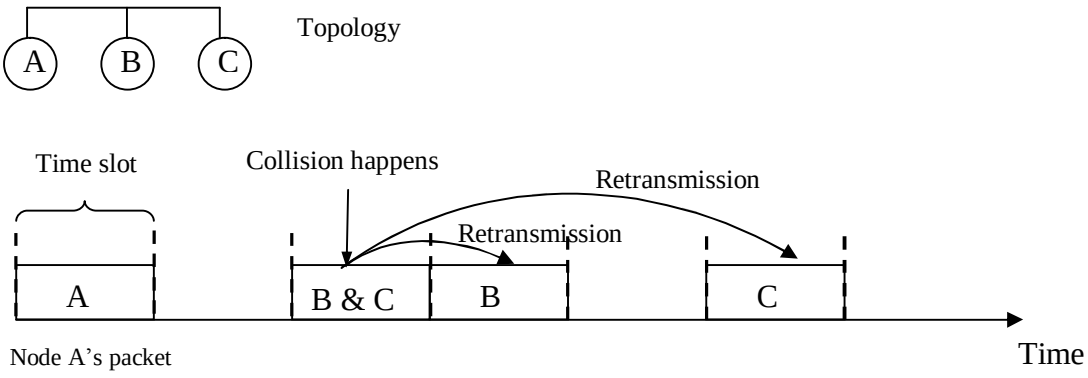


Figure 7. Collision mechanism in Slotted ALOHA

p-Persistent CSMA

Carrier Sense Multiple Access (CSMA) is the simplest form of medium access. In CSMA, a node who wants to send the packet will sense the channel and check if it is idle. If the channel is idle, it will send the packet immediately. If the channel is busy, it will perform a random back off by waiting before attempting to transmit again.

P-persistent CSMA is based on CSMA. In P-persistent CSMA, if the communication channel is not idle the node will perform a random back off. When the communication channel is idle, the node will send the packet with the probability P.

Luby's vertex coloring algorithm

Luby's algorithm [6, 7] is as Figure 8 described. In this algorithm, each node maps colors to timeslots and broadcasts in timeslots that have not been used by other nodes in its extended neighborhoods.

```
Let palette := colors; (* all colors *)
2 repeat
        c := choose(palette); (* tentative
*)
4       inform neighbors about c;
        if c was not chosen by a
neighbor then
6               output c; (* use c
permanently *)
```

Figure 8. Luby's algorithm [6, 7]

1.3 Problem definition

In [9], the authors take analytical approach to study the correctness of the algorithm and estimate its performance on different relocation scenarios in MANET. My work in this project is to study and validate this relocation analysis by simulating the algorithm on different mobility models. In order to generalize the relocation analysis, I create different mobility models in TOSSIM to reflect different scenarios in the real life. To further study how relocation issues affect the algorithm's performance. I simulate the algorithm considering different parameters of the relocation analysis (the maximum relocation ratio γ, the similarity ratio β and the relocation rate α). After the numerical

analysis of the collected data, I want to derive two functions to estimate the throughput by given the relocation parameters.

1.4 Document organization

This report consists of 4 chapters. Chapter 2 describes the Distributed Coloring Algorithm [9] and the Improved Distributed Coloring Algorithm [3]. Chapter 3 describes mobility models and numerical results. Two functions are deduced, one of them is used to estimate the throughput with a given pair of maximum relocation ratio and relocation rate, the other one is used to estimate the throughput with a given similarity ratio. Chapter 4 includes conclusions based on numerical results and analyses.

2 STUDIED ALGORITHMS

This chapter talks about the MAC algorithms I am going to study in this dissertation. One is called Distributed Coloring Algorithm which is proposed in [9]. The other one which is called Improved Distributed Coloring Algorithm is proposed in [3]. The Improved Distributed Coloring Algorithm is based on the Distributed Coloring Algorithm by adding several improvements, which is expected to achieve a better performance in MANET.

2.1 Distributed Coloring Algorithm

The Distributed Coloring Algorithm is proposed in [1]. In this algorithm, mobile nodes are able to learn some information from the success of the neighbors' broadcasts. Node always informs neighbors its broadcasting timeslot. So its neighbors could record this timeslot as occupied and do not choose this timeslot to broadcast. In this way, it prevents the collision that more than one node may choose the same timeslot to broadcast.

The structure of a communication round is illustrated as Figure 9 shows.

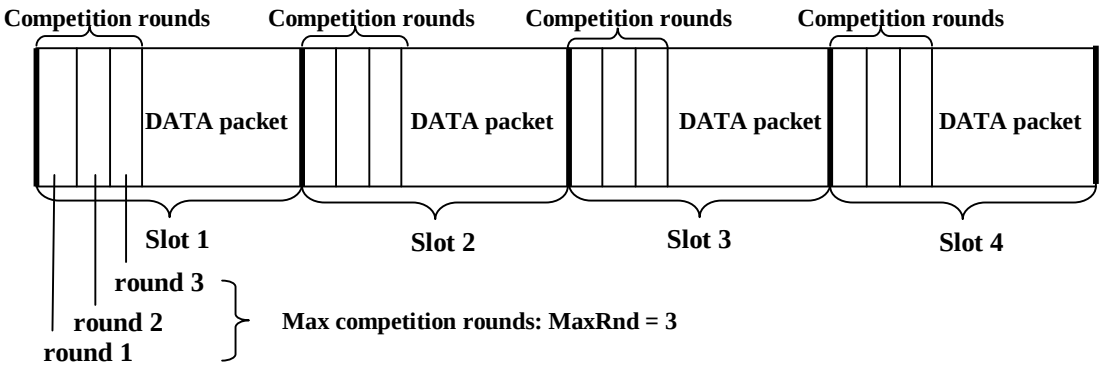


Figure 9 The construction of a broadcasting round

At the beginning of a communication round, for each node who wants to broadcast, it will try to acquire the communication channel first by following a competition strategy. All nodes who want to broadcast will come into the competition rounds first and become the competitors. During the competition rounds, each node participates in a certain round with a probability $p(k) = 2^{(-MaxRnd+k)}$, where $k \in [1;MaxRnd]$ is the round number of the competition. In case there are more than one competing nodes, there might be more than one winner after the competition. However, most of the competitors are expected to lose following this strategy. A successful broadcast indicates no

collision during the transmission period. This algorithm makes sure that there will be as little competitors as possible win the competition and broadcast eventually.

For simplicity, in this dissertation I call this Distributed Coloring Algorithm as the original algorithm. This algorithm is proved to converge in a finite period [9].

2.2 Improved Distributed Coloring Algorithm

Based on distributed coloring algorithm, the author in [3] suggests several ways to improve its performance especially due to the non-stationary situation. The improvements could be summarized as follows:

Improvement I
Node Pi marks each timeslot as **CONGESTED**, **EMPTY**, or **OCCUPIED**:

CONGESTED: when Pi detects reception error
EMPTY: when Pi detects nothing in that timeslot
OCCUPIED: when Pi successfully receives a data packet

When Pi decides to change its broadcasting time slot, it will choose one from timeslots marked by EMPTY with higher possibility than those marked by **CONGESTED**.

Improvement II
Simulated annealing algorithm: The famous Simulated Annealing (SA) is inspired from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. By analogy with this physical process, each step of the SA algorithm replaces the current solution by a random "nearby" solution, chosen with a probability that depends on the difference between the corresponding function values and on a global parameter T (called the temperature). For certain problems, simulated annealing may be more effective than exhaustive enumeration [5].

Inspired by simulated annealing, once a node Pi decides to choose a new timeslot marked by **CONGESTED**, Pi will get this timeslot with the probability exp(-(Si)/T) (Si: maximum energy level on the channel of last round; T: temperature). Otherwise Pi does not change its timeslot.

Improvement III
Inspired by P-persistent CSMA (Section 1.2.2), within a communication round Rj, if node Pi wants to broadcast, it will compete for the channel with the probability P,

otherwise it maintains silent. This strategy helps to accelerate the distribution of empty time slots at the booting period of the network and when relocation happens frequently.

Improvement IV

Let a node Pi check its broadcasting timeslot change or not in the beginning of a new communication round. If this timeslot has not been changed for several rounds, let Pi reduce its competition rounds when it comes into a new communication round. This strategy makes sure that once the communication already becomes stable, then the competition should be less required since each node has been assigned an available timeslot. In this way, the overhead of the network will be reduced.

Improvement V

Let a node Pi check the states of all timeslots in the beginning of a communication round. If Pi finds that timeslots marked by EMPTY never decrease for some time, which means there are always enough timeslots for all competitors to broadcast, it will take one of them directly without the competition. This strategy makes sure that if the number of empty timeslots is larger than the number of competitors, the competition is not needed. In this way, it can reduce the overhead of the network.

To sum up, Improvements I-III are used to shorten the stabilization time. Improvements IV and V are used to reduce the overhead.

For simplicity, in this dissertation I call this Improved Distributed Coloring Algorithm [2] as the improved algorithm.

3 MOBILITY MODELS AND ANALYSIS

In order to validate the relocation analysis of these two algorithms for MANET, I build several mobility models in software environment to reflect the real world's different mobility scenarios. In the real world, nodes do not move completely random, their movements still follow certain rules. For example, cars can only run on the road, people who are shopping or working act in certain area, birds usually stay close to each other and keep moving as a group, etc. So choosing appropriate mobility models which could reflect these attributes is very important. I choose mobility models by following two principles: reflect real world's scenarios as much as possible and can be implemented in the simulation environment as simple as possible.

I considered several mobility models which could depict the real world's characteristics. After the comparison, finally I choose four typical mobility models and simulate both the original and the improved algorithms on them to validate the relocation analysis.

3.1 Platform and tools

The original and the improved algorithms are implemented in TinyOs, which is written in nesC programming language. The mobility models are built in TOSSIM, which is written in Python programming language.

TinyOs

TinyOs is a component-based operating system which is used for wireless sensor network (WSN). It is written in nesC programming language and is an embedded operating system. The reason that it is optimized for WSN is because its advantages in energy consuming and memory limitations.

TinyOs programs are composed of software components. These components could be seemed as hardware abstractions. Components are connected to each other with interfaces. These interfaces and components are used for describing packet communication, routing and sensing etc. In this project, we use TinyOs 2.1.x as the platform.

TOSSIM

TOSSIM is an in-built simulator for TinyOS, which offers a simulation environment for users to debug, test, and analyze algorithms.

TOSSIM is an event-based simulator. Events are arranged in a queue sorted by time. When it runs, events will be pulled from the queue and executed. These simulation events can represent packet transmissions or hardware interrupts. It takes a short time (e.g., a few microseconds) to run a task.

3.2 Terminology

Extended neighborhood

We call Node i's neighborhood and all node i's neighbors' neighborhood together as node i's extended neighborhood. For example, if Node A's neighbors are {B, C} and Node A's neighborhood is N_A. Suppose N_B and N_C are Node B's and Node C's neighborhoods. Then N_A, N_B and N_C are Node A's extended neighborhood.

Relocation rate α

The relocation rate depicts how many nodes in the network will relocate from their original position between two steps of the algorithm. The number of nodes who relocate from their original position divided by the number of all nodes is defined as this mobility model's relocation rate α, $\alpha \in [0, 1]$.

$$\boldsymbol{\alpha} = \mathbf{Nodes}_{\text{relocated}} / \mathbf{Nodes}_{\text{all}} \quad (1)$$

Similarity ratio β

The similarity ratio depicts changes of a node's extended neighborhood once it relocates. In our implementation, a node will share the communication channel with all its extended neighbors. Let us consider a mobile node pi, which relocates from its old extended neighborhood Ni, to a new one Ni'. We assume that a ratio of β extended neighbors of pi in its current extended neighborhood is going to be in its new extended neighborhood, i.e., $\beta \leq |Ni \cap Ni'| / |Ni'|$, where $\beta \in [0,1]$ is named the similarity ratio.

$$\boldsymbol{\beta} = \mathbf{Extended\_Neighbors}_{\text{old}} / \mathbf{Extended\_Neighbors}_{\text{new}} \quad (2)$$

Maximum relocation ratio γ

For a node pi, the maximum relocation ratio is defined as the maximum relocation distance of pi divided by the width (or the length) of the network topology (In this dissertation, the length equals to the width for all topologies).

3.3 Constrained-jump model

The Constrained-jump model is used to depict nodes' movement within a certain area. In this model, nodes are randomly distributed in the simulation area in the beginning, as Figure 15 shows.

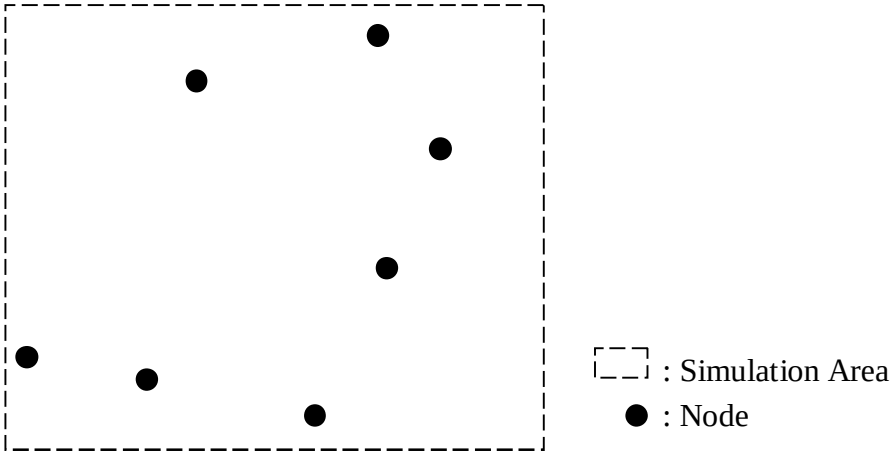


Figure 15. The distribution of nodes in Constrained-jump model

At certain time, a relocating node will randomly choose a target T within a certain area and jumps to that target, as Figure 16 shows.

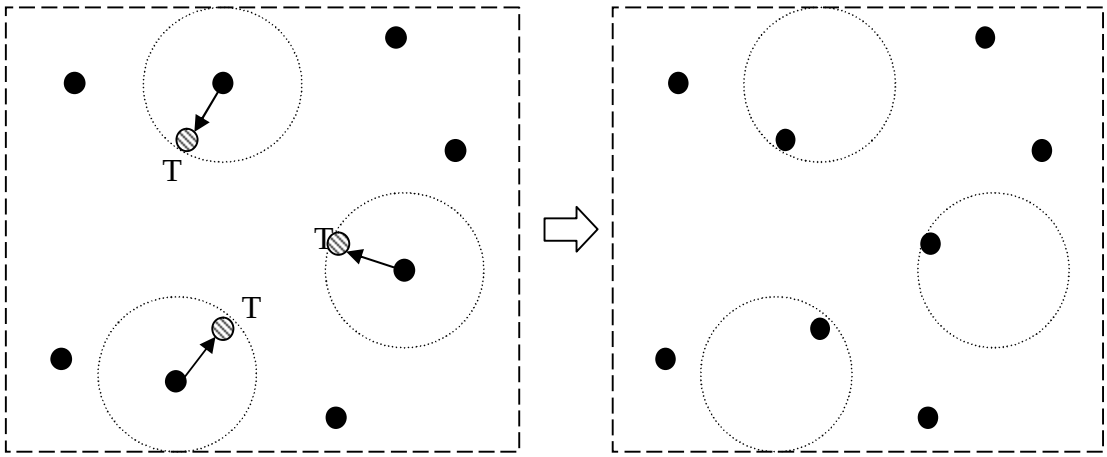


Figure 16. How the Constrained-jump model works

In Constrained-jump model, I put 400 nodes which are randomly distributed on a 200× 200 area. Figure 17 shows the numerical results of the original algorithm and the improved algorithm on Constrained-jump model. I simulate both algorithms under 40 different pairs of maximum relocation ratios and relocation rates. The X-axis is the relocation rate while the Y-axis is the maximum relocation ratio; the figure shows the throughput under these parameters.

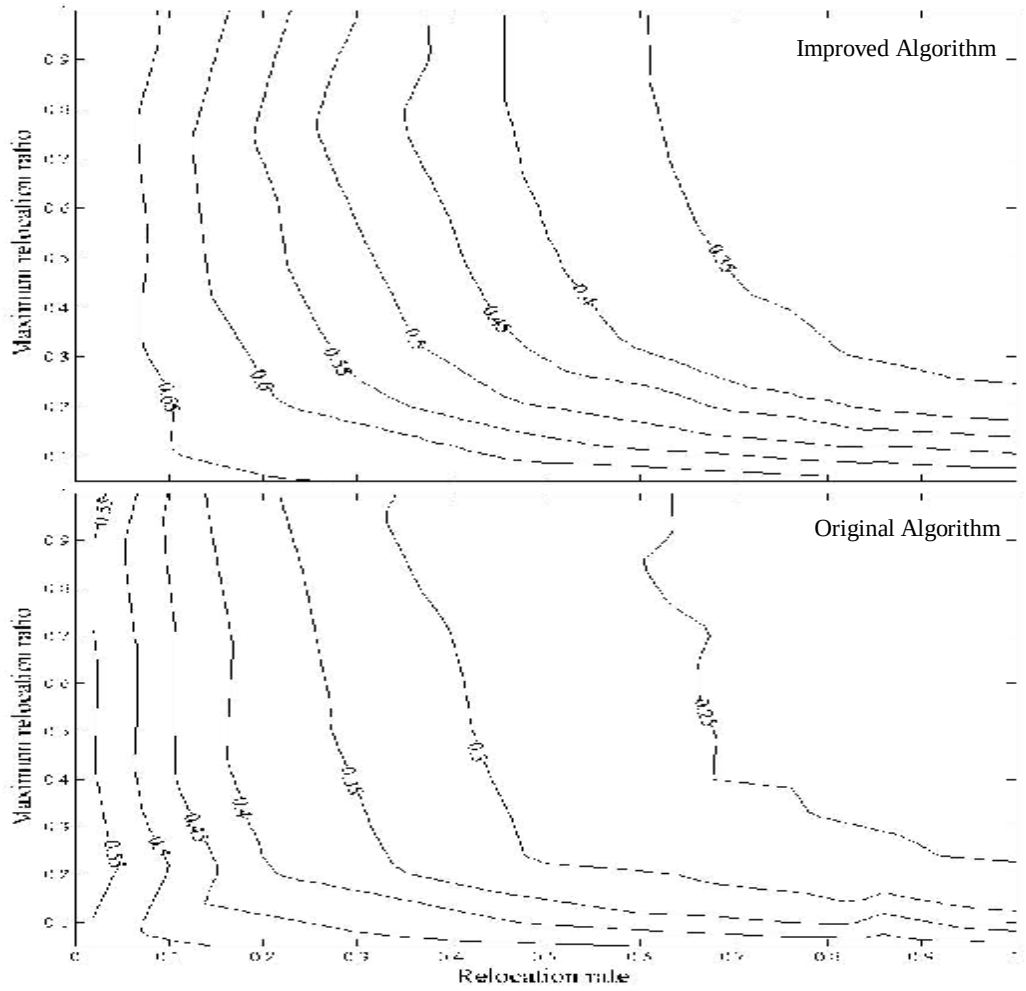


Figure 17. Throughputs of original and improved algorithms on Constrained-jump model with different relocation rates and different maximum relocation ratios.

The improved algorithm always achieves a higher throughput at each pair of maximum relocation ratio and relocation rate, which is decided by the advantages of the improved algorithm mentioned in section 2.2. By checking the definition of maximum relocation ratio (section 3.1), we know that the algorithm achieves a higher throughput with a lower maximum relocation ratio is because lower maximum relocation ratio means less jump distance when nodes relocate. According to the algorithms [1, 3], a node will share the same communication channel with its extended neighbors. Changes to the extended neighborhood will cause a collision of an already-established communication, which reduces the throughput.

Considering the same algorithm under the same maximum relocation ratio, the reason that the throughput decreases as the relocation rate increases is because higher relocation rate means more nodes are relocating. With a certain maximum relocation

ratio, higher relocation rate causes more changes of nodes' neighborhood in the entire network, which will affect the network's throughput.

Both algorithms' throughputs change faster with lower maximum relocation ratio (<0.2) and lower relocation rate (<0.3). Once the relocation rate and the maximum relocation ratio become higher, the throughput changes slower. The reason that the throughput changes slower with high relocation rates is because in this algorithm, mobile nodes are able to learn some information from the success of neighbors' broadcasts and make use of such information to share the communication channel. When there is a high relocation rate, such information becomes useless since too many nodes will relocate, which causes nodes' neighborhoods changing all the time. The reason that the throughput changes slower with high maximum relocation ratio is because the size of a node's extended neighborhood is a constant, which means no matter how far a node will jump away, once the jump distance is larger than the size of this node's extended neighborhood, this node will always jump to a entire new extended neighborhood and rebuild the connection with all its new extended neighbors.

In order to further analysis the relationship between the relocation parameters, I also test the similarity ratios of the mobility model at each pair of maximum relocation ratio and relocation rate. Figure 18 shows the similarity ratio of the improved algorithm at each pair of maximum relocation ratio and relocation rate.

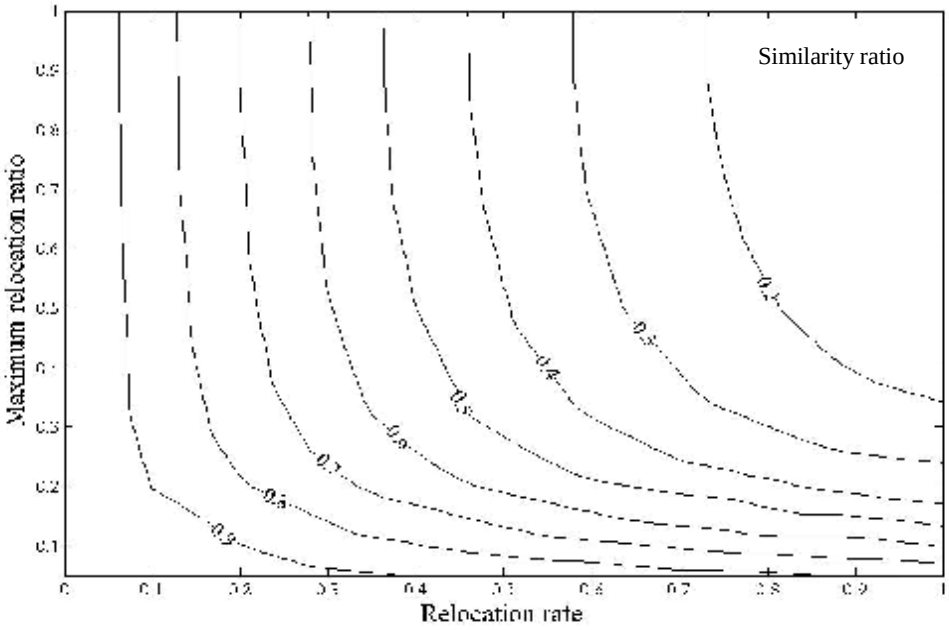


Figure 18. Similarity ratio at each pair of maximum relocation ratio and relocation rate

From Figure 18 we may find that each pair of maximum relocation ratio and relocation rate can uniformly determine a similarity ratio. The similarity ratio is high with low

maximum relocation ratio and relocation rate, which is decided by the definition of maximum relocation ratio and relocation rate (Section 3.2). To study the relationship between the throughput and the similarity ratio, I get the throughput at each similarity ratio, as Figure 19 shows. The X-axis is the similarity ratio while the Y-axis is the throughput of the algorithm.

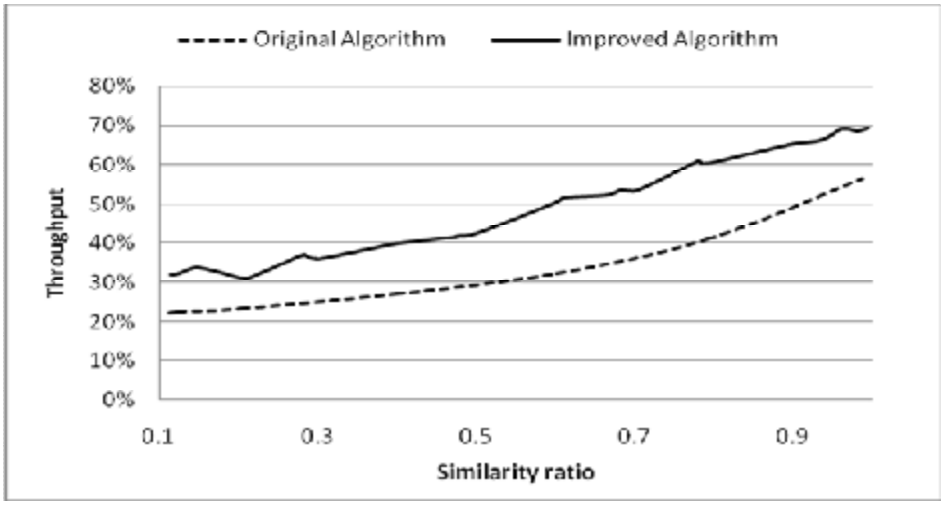


Figure 19. The relationship between the throughput and the similarity ratio

From this figure we can see that the improved algorithm always achieve a higher throughput than the original algorithm at each similarity ratio. Both algorithms' throughputs increase as the similarity ratio increases. Checking the definition of similarity ratio (section 3.1), the similarity ratio depicts the changes of a node's extended neighborhood. A node will share the same communication channel with its extended neighbors. Lower similarity ratio means more changes of a node's extended neighborhood due to its relocation activity, which will cause a collision of an already-established communication and reduce the throughput.

Now, I have showed the throughput under different relocation parameters. The next step is to further study this numerical relationship and derive functions based on this relationship, which can be used to estimate the throughput in more complex mobility models.

3.4 Expectation functions based on Constrained-jump model

3.4.1 The motivation

In Constrained-jump model, I simulate the improved algorithm under many pairs of maximum relocation ratio and relocation rate. I also simulate the algorithm under different values of similarity ratio. These data reflect the relationship between the

throughput and different parameters of the relocation analysis: the relocation rate α, the maximum relocation ratio γ and the similarity ratio β. By analyzing these data, I am able to derive the functions T(γ, α) and T (β), which can be used to estimate the algorithms' throughputs. T(γ, α) is used to predict the throughput by given a maximum relocation ratio γ and a relocation rate α. T (β) is used to predict the throughput by given a similarity ratio β.

3.4.1 Expectation function based on relocation rate and maximum relocation ratio

In order to make the expectation function more accurate, I test 50 more pairs of maximum relocation ratios and relocation rates. In Matlab, using fifth degree fitting function, the following fitting graph (Figure 20) and fitting function (3) can be generated.

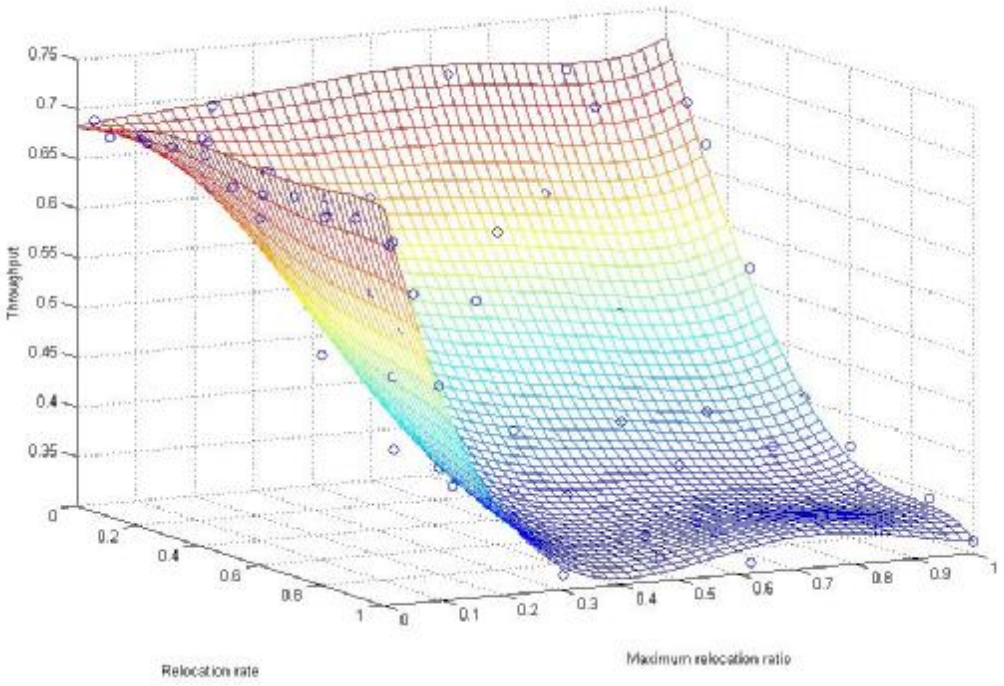


Figure 20. The fitting graph based on fifth degree polynomial function

$$T(\gamma, \alpha) = 0.6826 - 0.0943*\alpha + 0.0100*\gamma - 0.4726*\alpha^2 - 3.8276*\alpha*\gamma + 0.7574*\gamma^2 + 1.8610*\alpha^3 + 5.5476*\alpha^2*\gamma + 0.9655*\alpha*\gamma^2 - 2.7558*\gamma^3 - 2.8114*\alpha^4 - 3.7688*\alpha^3*\gamma - 0.4291*\alpha^2*\gamma^2 + 1.0848*\alpha*\gamma^3 + 3.3897*\gamma^4 + 1.3663*\alpha^5 + 1.0474*\alpha^4*\gamma - 0.3064*\alpha^3*\gamma^2 - 0.0145*\alpha^2*\gamma^3 - 0.722*\alpha*\gamma^4 - 1.3845*\gamma^5 \quad (3)$$

With a given pair of maximum relocation ratio γ and relocation rate α, T(γ, α) is able to calculate the expectation of the throughput.

3.4.2 Expectation function based on similarity ratio

In excel, I use the fifth degree polynomial fitting function to derive the expectation function T(β) (4) based on different similarity ratios, as Figure 21 shows.

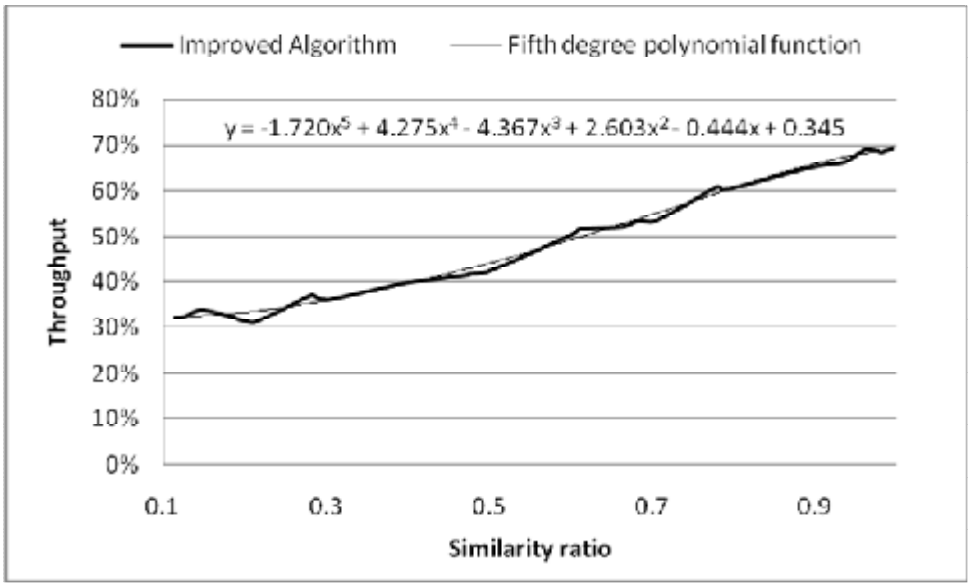


Figure 21. The fitting graph based on fifth degree polynomial function

$$T(\beta) = -1.720*\beta^5 + 4.275*\beta^4 - 4.367*\beta^3 + 2.603*\beta^2 - 0.444*\beta + 0.345 \quad (4)$$

With a given value of similarity ratio β, T (β) is able to calculate the expectation of the throughput at that similarity ratio.

In order to further validate the relocation analysis and check the performance of these two fitting functions, I create two more mobility models: Parallel-motion and Flock models. I test expectation functions on these two models and compare their performance.

3.5 Parallel-motion model

The Parallel-motion model is used to depict cars' movement in highway. Figure 22 illustrates how this model works. In the beginning, nodes are distributed in a matrix. I divide these nodes into odd lines and even lines. Nodes in the even line will move to the same direction with the same speed v. Take node i as an example, in this model there will be nodes coming in and out its neighborhood all the time, which means the topology of the network will never be stable. I simulate the improved algorithm on this model to see how it performs and verify the performance of the expectation functions.

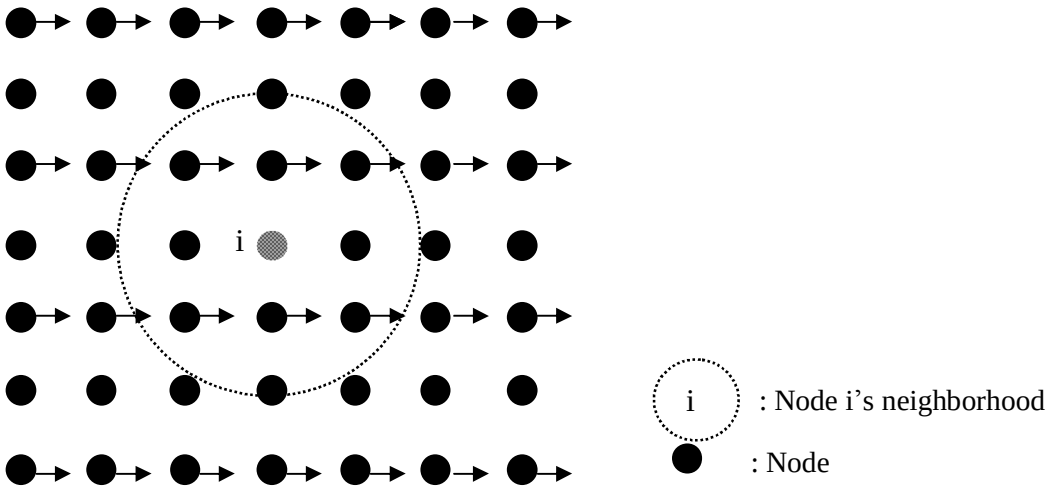


Figure 22. Parallel-motion model

In this model, I put 400 nodes distributed on a 200×200 area with a matrix distribution. The distance between each two nodes is 10. The radius of node i's neighborhood is 11 and the radius of node i's extended neighborhood is 22. Since in this model, the even lines keep moving while the odd lines keep still, which means the relocation rate α is a constant 0.5. I simulate the improved algorithm with two maximum relocation ratios: 2.5% and 5%.

Figure 23 illustrates the throughputs of the improved algorithm among the communication rounds. The X-axis is the communication round while the Y-axis is the throughput.

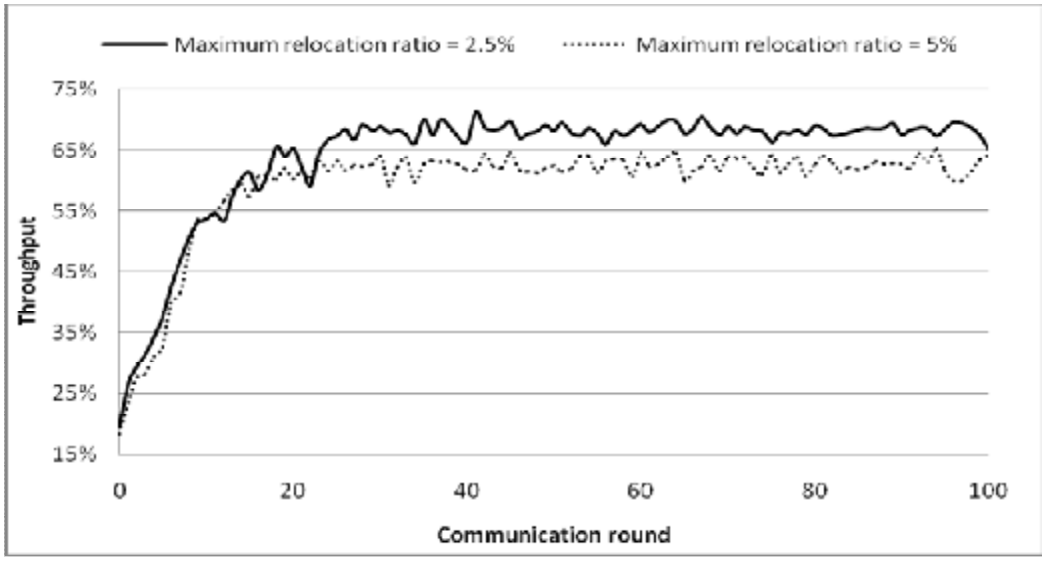


Figure 23. Throughput of the improved algorithm with different maximum relocation ratios

Comparing throughputs with different maximum relocation ratios, we could find that the algorithm under a lower maximum relocation ratio could achieve a higher

throughput. Checking the definition of maximum relocation ratio, it is easy to understand that lower maximum relocation ratio causes less changes of the network topology. According to the algorithm [1, 3], network could achieve a higher throughput due to less changes of the topology.

Figure 24 shows the expected value of the throughput based on the function T(γ, α).

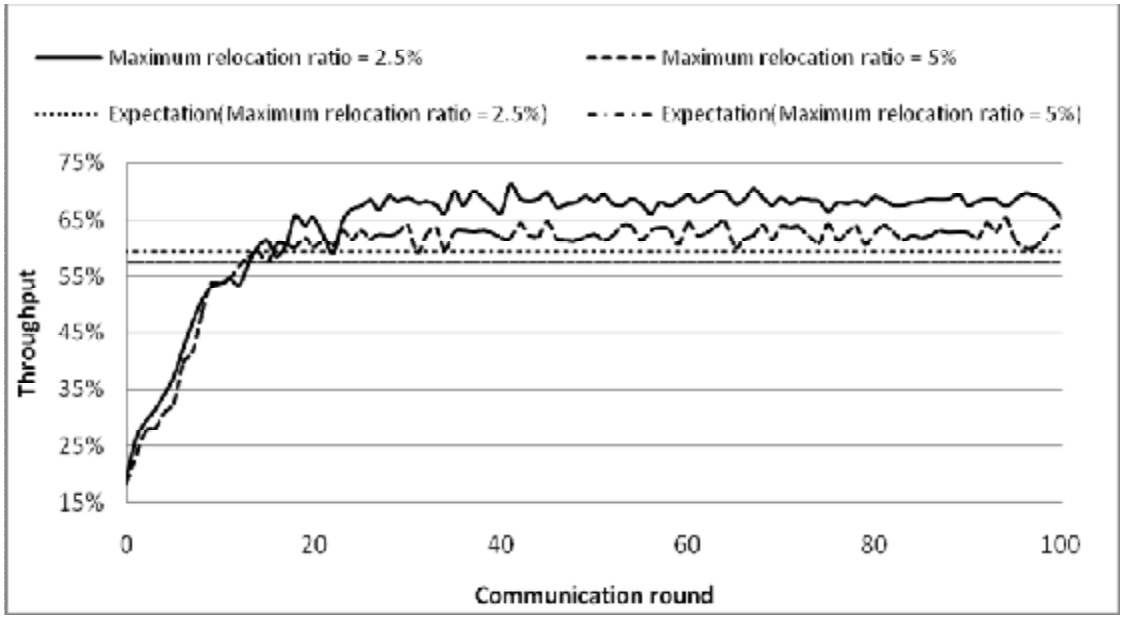


Figure 24. Expected throughputs of the improved algorithm based on T(γ, α).

The reason that there is a big difference before the 20th communication round is because the network is not stable yet and T(γ, α) is derived based on values collected when the network is stable.

I also test the similarity ratio among the communication round, as Figure 25 shows.

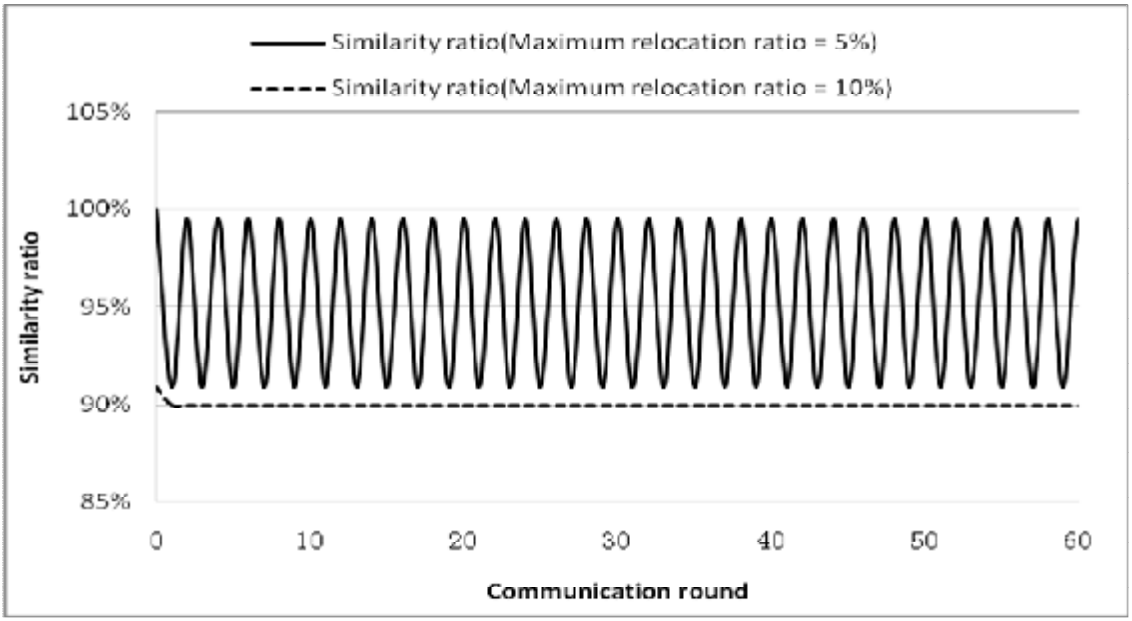


Figure 25. Similarity ratios among the communication round

From Figure 25 we may find that the similarity ratio with speed 5 is higher than speed 10, which causes the throughput under speed 5 is higher than speed 10.

Figure 26 shows the expected value of the throughput based on the function T (β).

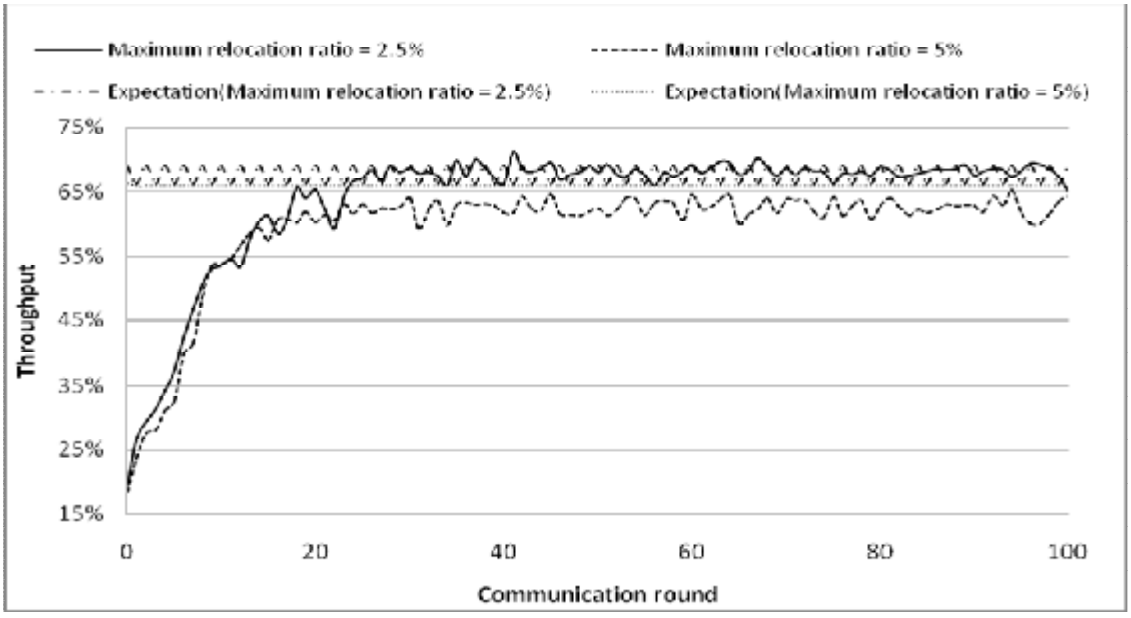


Figure 26. Expected throughputs of the improved algorithm based on T (β)

In order to compare the performance of functions T(γ, α) and T (β). I calculate the difference between the predicted values and the numerical values when the throughput becomes stable (started from 25th communication round), as Figure 27 shows.

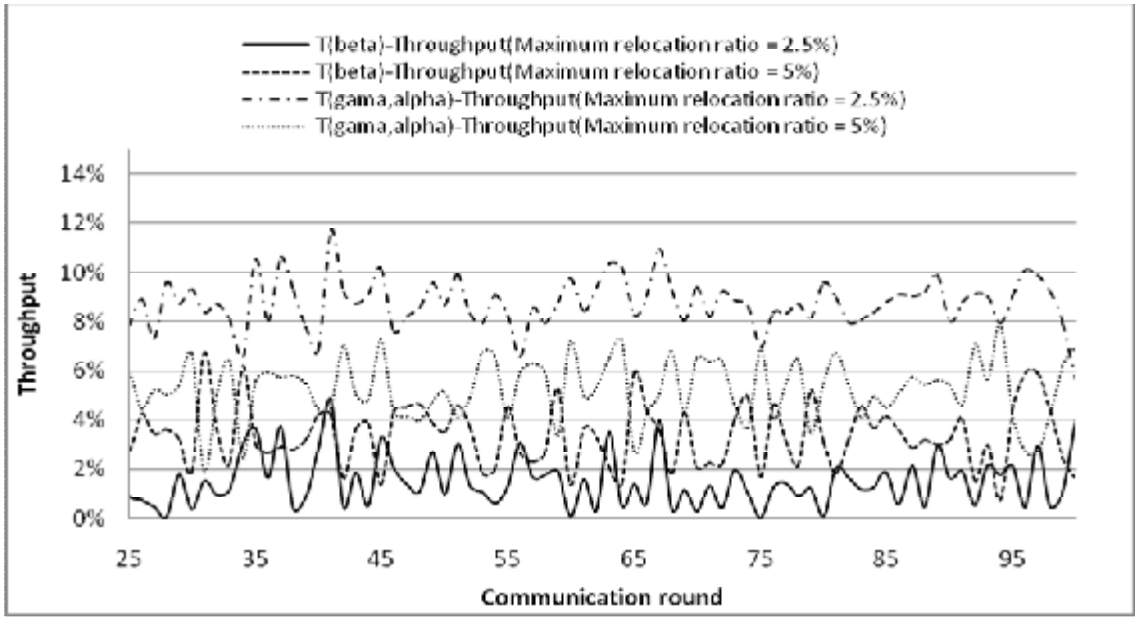


Figure 27. Difference between the expected throughputs and the numerical throughputs

From Figure 27, we may find that the expectation based on T (β) gives a better performance than the expectation based on T(γ, α).

3.6 Flock model

Nowadays, sensor nodes have been broadly applied on animals to collect data and study their social behavior, in combination with microclimate conditions, to protect the animals' habitat and ensure their well-being. In order to get the performance of the improved algorithm on this purpose, I create the Flock model to model the simple behavior of animals, such as birds, whales and zebras that usually stay with each other as a flock.

First of all, we need to figure out what kind of characters does a flock has. Here I use birds as an example. A flock's characters could be summarized as followed:

- **Separation:** Steer to avoid being too close to each other.
- **Alignment:** Steer towards the average heading of other birds in the flock.
- **Cohesion:** Steer to move toward the average position of birds in the flock.

Then I create the model of a bird as Figure 28 shows. The vertex points the direction of the bird. A direction and a speed could decide where this bird goes to. We need two ranges to monitor the mutual behavior among birds. One is called Detection Range,

which defines the detection range. A bird will detect another bird only when another bird is within this range. As a bird, it will adjust its moving direction by considering all the other birds' positions it detects. The other range is called Separation Range which depicts the minimum distance between two birds. With this Separation Range, positions of birds will not be overlapped. The color of a bird stands for its group. Different colors mean different groups.

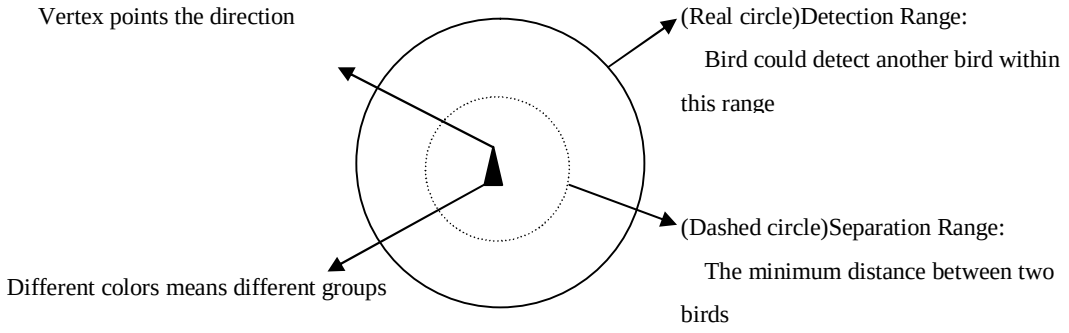


Figure 28. The model of a bird

After we created the model, next step is considering how to depict features of a flock.

- **Simulate the Separation**

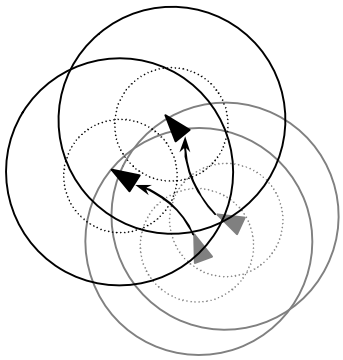


Figure 29. The Separation of birds

As Figure 29 shows, when two birds of the same color (same group) stay too close to each other (within the separation range), they will repel each other.

- **Simulate the Alignment**

As Figure 30 shows, when two birds of the same color (same group) detect each other, they will adjust their directions and move forward together.

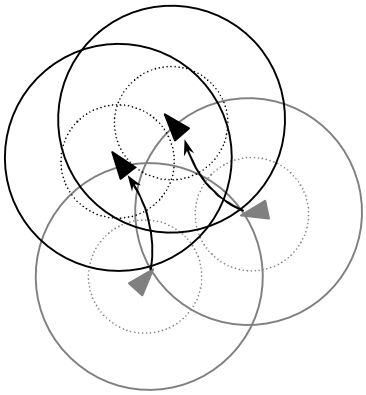


Figure 30. The Alignment of birds

Ÿ Simulate the Cohesion

A bird will always intend to move to the average position among all the other birds it detected, as Figure 31 shows. In this way, a flock could stay tight.

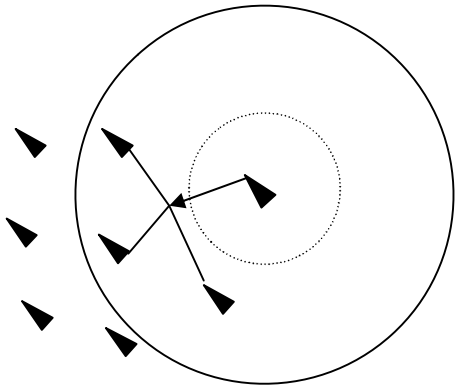


Figure 31. The Cohesion of birds

After considering all these features, I create the Flock model as Figure 32 describes. I put two groups of birds and each of them stays at one side of the simulation area. I let both groups running for a certain time until the network becomes stable. Then these two groups will move face to face and meet in the middle and separate from each other later.

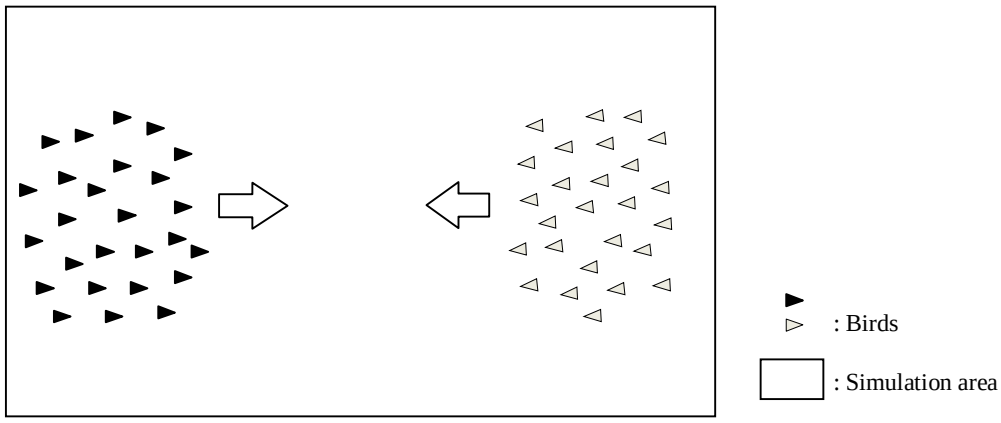


Figure 32. The Flock model

In this model, I put two groups of birds in a 400×200 area. Each group has 100 birds distributed in a 10×10 matrix at one side of the simulation area. The separation range is 10 and the detection range is 20. In the beginning, I let both groups run for a certain period until the network becomes stable. Then two groups will move toward the same direction and meet in the middle. I get the throughput of the improved algorithm with two speeds: 5 and 10, as Figure 33 describes. The X-axis is the communication round while the Y-axis is the throughput of the network.

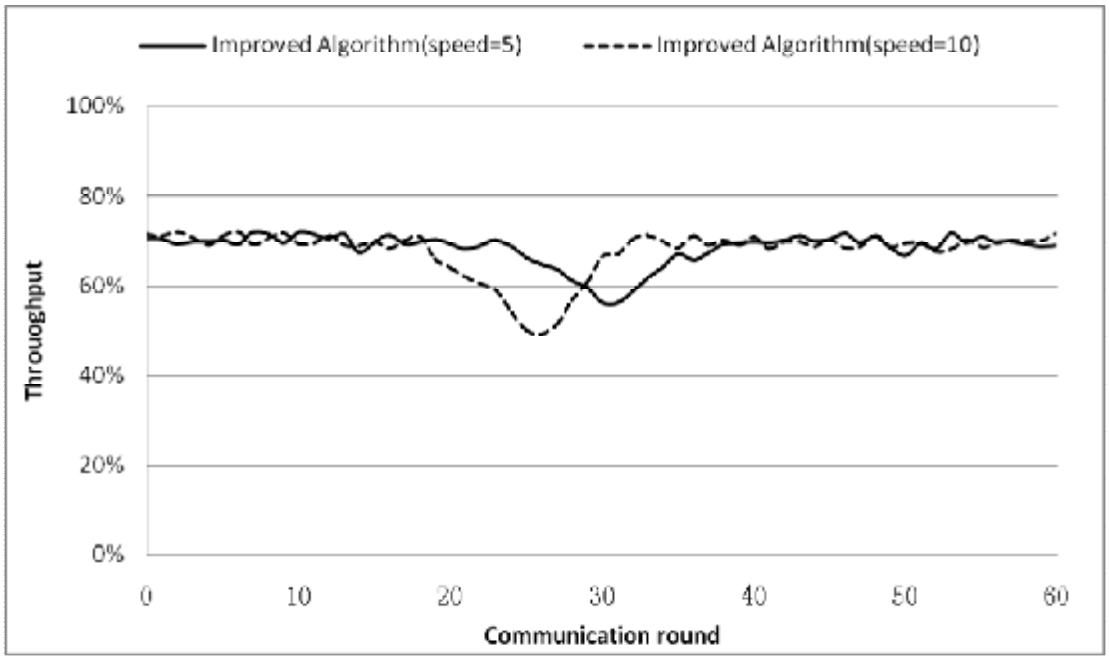


Figure 33. Throughputs of the improved algorithm with speed 5 and 10

As Figure 33 illustrates, the throughput is stable when both groups are stable. Take speed=10 as an example, around the 20th communication round there is a decrement for the throughput due to their meeting in the middle. Then the throughput increases until

stabilize again. The lowest throughput of the improved algorithm with speed 5 is higher than the one with speed 10.

In order to understand the variation of throughputs, I calculate the similarity ratio β among the communication round. Here, I need to redefine the relocation rate and similarity ratio since there is a special case in this model, which happens when two groups meet in the middle. In this model, as a node, its extended neighbors are either in the same group or in the other group. When they meet in the middle, nodes in the same group will never relocate after the stabilization period while nodes from the different group will always relocate.

I redefine the relocation rate in this special case. When these two groups meet in the middle, for each node i, I redefine the relocation rate $\alpha_{\text{Node i}}$ as the number of extended neighbors from the different group divided by the size of the group, as formula (3).

$$a_{Nodei} = \frac{ExtendedNeighbor_{different}}{n} \tag{5}$$

Then I calculate the average relocation rate as the relocation rate of the entire network α_{entire} as formula (4) describes.

$$a_{entire} = \frac{\sum_{i=0}^{n} a_{Nodei}}{n} \tag{6}$$

Node i's similarity ratio $\beta_{\text{Node i}}$ is defined as the number of extended neighbors in the same group divided by the number of all its current extended neighbors (5) and β_{entire} is defined as formula (6).

$$b_{Nodei} = \frac{ExtendedNeighbor_{same}}{ExtendedNeighbor_{current}} \tag{7}$$

$$b_{entire} = \frac{\sum_{i=0}^{n} b_{Nodei}}{n} \tag{8}$$

Figure 34 illustrates similarity ratios of the network under two speeds 5 and 10. The X-axis is the communication round while the Y-axis is the similarity ratio of the network.

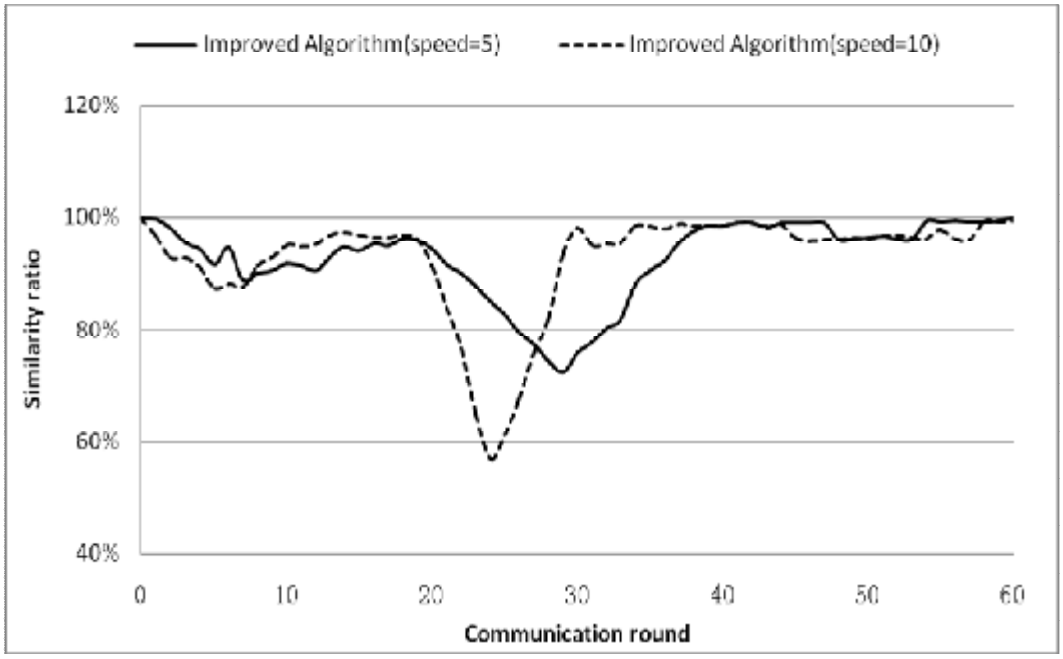


Figure 34. Similarity ratios of the Flock mobility with two speeds 3 and 6

From Figure 34 we can clearly find that the similarity ratio has a similar variation as the throughput among the communication round. When the similarity ratio is stable, the throughput is stable. When two groups meet in the middle, the similarity ratio also decreases. The similarity ratio with a higher speed decreases more than the one with a lower speed, which causes the throughput with the lower speed is higher than the one with the higher speed.

Figure 35 shows the expected throughput T(β) based on these two speeds.

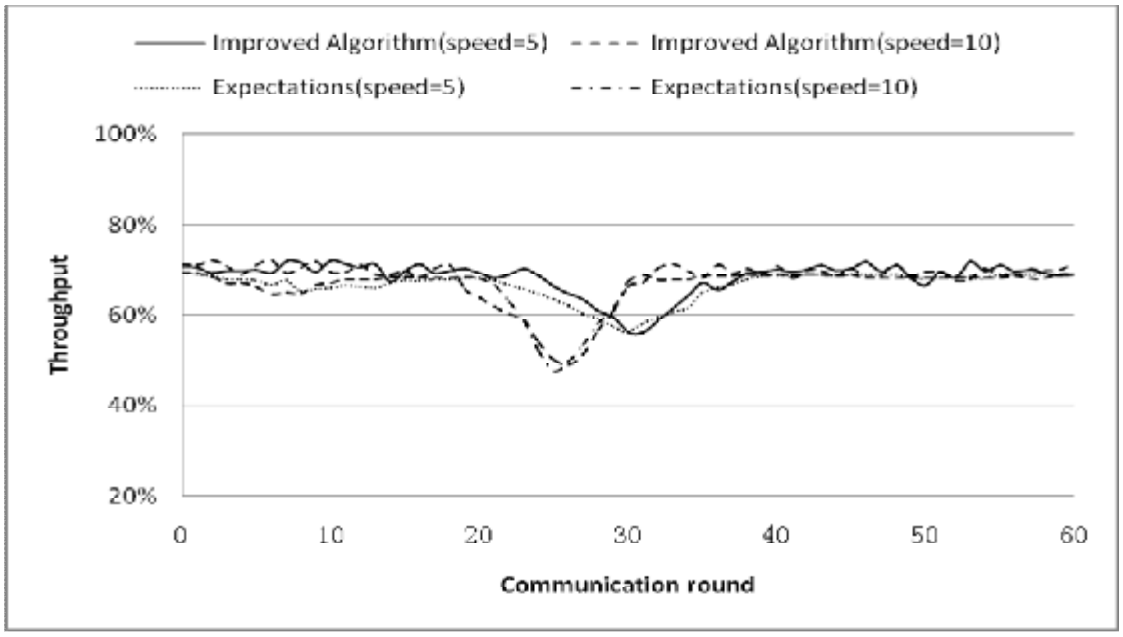


Figure 35. The expectation of throughput based on T(β)

In order to compare the performance of expectation functions T(β) and T(γ, α). I also calculated the expected throughput with T(γ, α). Since I test two different absolute speeds 5 and 10 and two groups move towards the same direction, which means the relative speeds should be 10 and 20. So the maximum relocation ratios are 10% and 20% according to the definition, predicted throughputs are as Figure 36 shows.

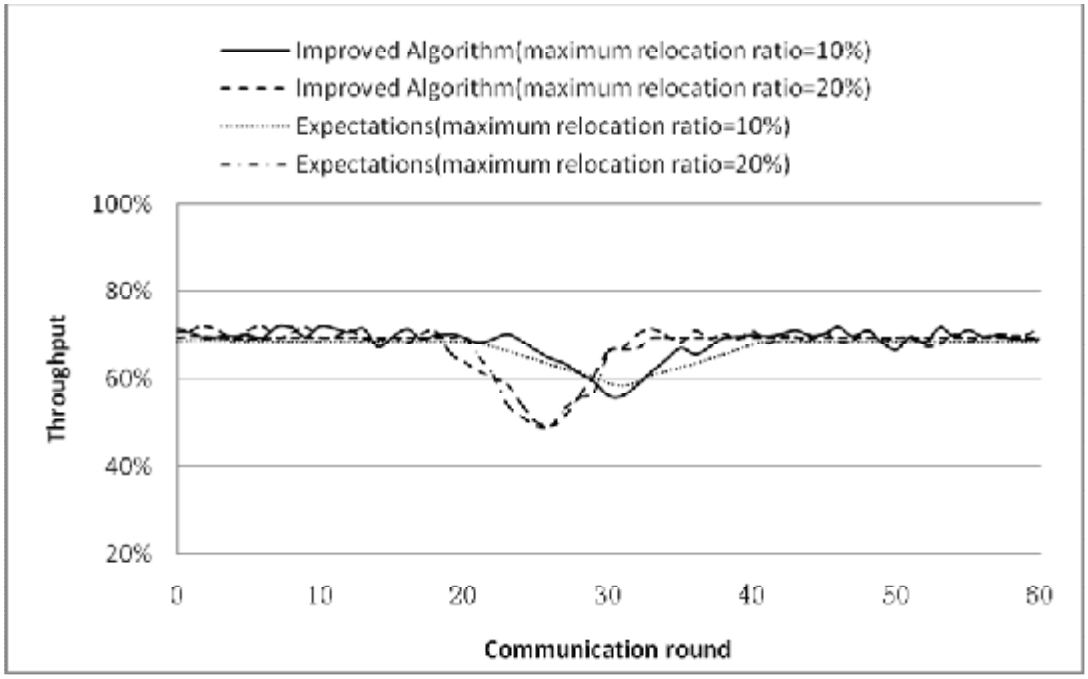
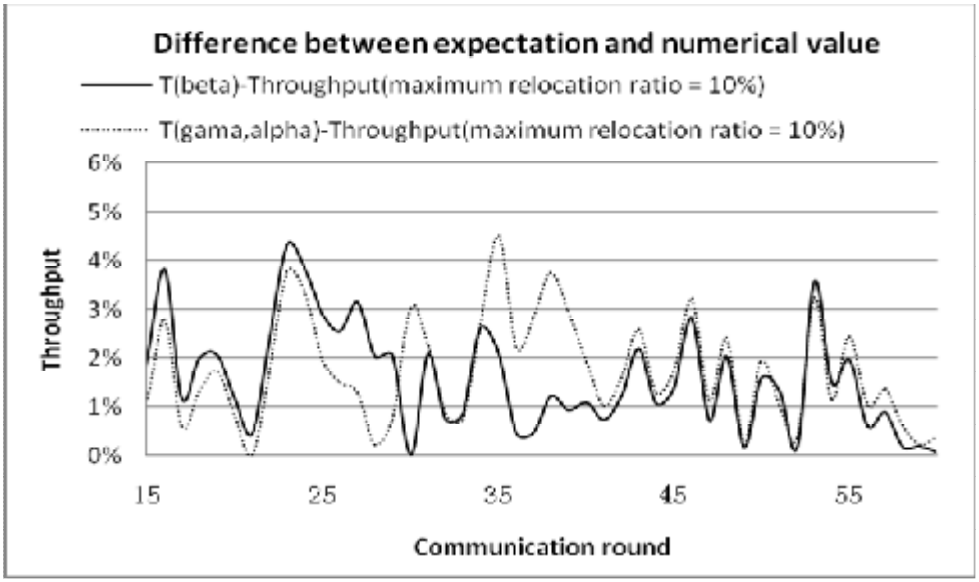


Figure 36. The expectation of throughput based on T(γ, α)

To compare the performance of functions T(γ, α) and T (β). I calculate the difference between the expected values and the numerical values (started from 15th communication round), as Figure 37 shows.



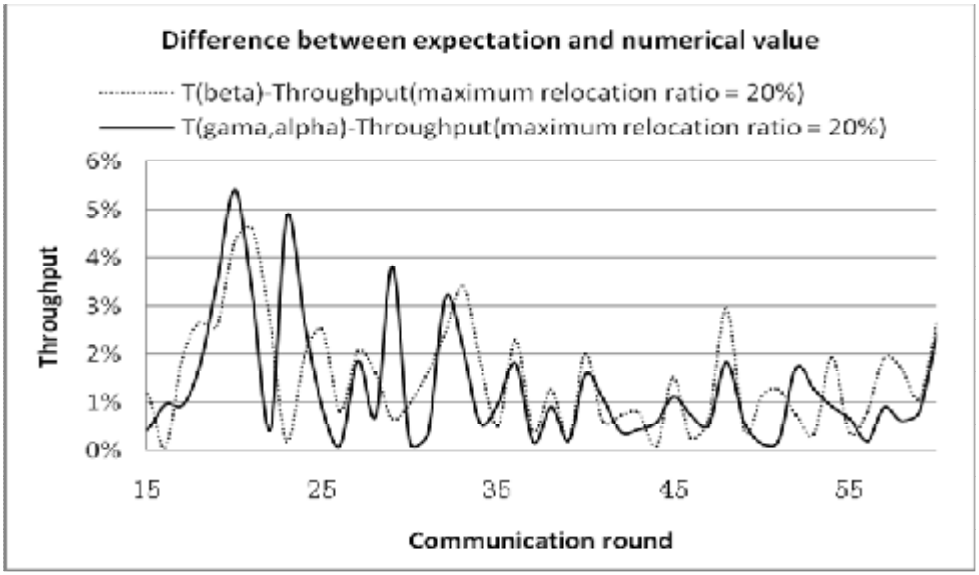


Figure 37. Difference between the expected throughputs and the numerical throughputs

Comparing Figure 36 and Figure 37, we may find that functions T(β) and T(γ, α) achieve an equally good performance predicting the throughput.

4 CONCLUSIONS

This dissertation presents three mobility models: the Constrained-jump, Parallel-motion and Flock models. The first model is mainly used for comparing the performances of the two MAC algorithms. From the comparison, we may find that the improved algorithm performs better than the original algorithm. Two functions $T(\gamma, \alpha)$ and $T(\beta)$ are derived from these numerical results. $T(\gamma, \alpha)$ estimates the algorithm's throughput for a given pair of maximum relocation ratio γ and relocation rate α. $T(\beta)$ estimates the algorithm's throughput for a given similarity ratio β. The two latter models, Parallel-motion and Flock, validate the functions $T(\gamma, \alpha)$ and $T(\beta)$. It turns out that throughputs of the Parallel-motion model can be better estimated using the function $T(\beta)$, while throughputs of the Flock model can be estimated well using both functions $T(\gamma, \alpha)$ and $T(\beta)$.

5 REFERENCES

[1] Christian Carling, Pontus Svenson, Christian Mårtenson, Henrik Carlsen, "A Flock-Based Model for Ad Hoc Communication Networks, in Cd Proceedings of the Eight International Command and Control Research and Technology Symposium", Washington, DC, USA, 17-19 June 2003.

[2] Chane Lee Fullmer and J. J. Garcia-Luna-Aceves. "Solutions to hidden terminal problems in wireless networks" in: SIGCOMM, pages 39–49, 1997.

[3] Gongxi Zhu. "Evaluating New MAC algorithms for MANET (Mobile Ad-Hoc Network)" Master thesis, Feb.2010.

[4] Hideaki Takagi and Leonard Kleinrock. "Throughput analysis for persistent CSMA systems" in: IEEE Transactions on Communications, 33(7):627–638, Jul 1985.

[5] Janez Žerovnik. "On the convergence of a randomized algorithm frequency assignment problem" in: Central European j. operat. resear. econom., 1998, vol. 6, no. 1-2, p. 135-151.

[6] Michael Luby (1986). "A simple parallel algorithm for the maximal independent set problem" in: SIAM J.Comput. 15(4):1036-1053

[7] Michael Luby (1993). "Removing randomness in parallel computation without a processor penalty" in: J.Comput.Syst.Sci. 47(2):250-286

[8] Norman Abramson. "Development of the ALOHANET" in: IEEE Information Theory, 31(2):119–123, 1985.

[9] Pierre Leone, Marina Papatriantafilou and Elad Michael Schiller, "Relocation Analysis of Stabilizing MAC Algorithms for Large-Scale Mobile Ad Hoc Networks" in: ALGOSENSORS, 2009. Also appears as a technical report 2008:23, Department of Computer Science, and Engineering, Chalmers University of Technology (Sweden), Sep. 2008.

[10] Philip Levis, Nelson Lee, Matt Welsh and David Culler. "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications" in: ACM Conference on Embedded Networked Sensor Systems, 2003.