2012

Cooperative Computing Techniques in Networked Systems

Phisan Kaewprapha
Lehigh University

COOPERATIVE COMPUTING TECHNIQUES IN NETWORKED SYSTEMS

by

Phisan Kaewprapha

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

Electrical Engineering

Lehigh University

September, 2012

Approved and recommened for acceptance as a dissertation in partial fulfillment of the requirements for the degree of
Doctor of Philosophy.

(Date)

(Accepted Date)

Tiffany J. Li,PhD.
Chair and Advisor

Zhiyuan Yan, PhD.
Committee Member

Shalinee Kishore, PhD
Committee Member

Liang Cheng, PhD
Committee Member (External)

Acknowledgements

I have always been receiving great supports from my advisor, Prof. Tiffany Jing Li. And that have always been warm and positive ones. I would like to get this chance thank for her kindness and all the knowledge I have learned since I started my educational journey at Lehigh University.

It has been no doubt that I have also had a great experiences from all other professors I have taken classes with, especially Prof. Meghanad D. Wagh, Prof. Zhuyuan Yan, Prof. Shalinee Kishore and Prof. Liang Cheng who some also are my committee members. They have given me wise advices, feedbacks and discussion regarding my dissertation.

I would also like to thank for the financial support from the Thai Government and their staffs for providing support and guideline throughout the program.

I would like to thank my all my friends and colleagues for the great experience sharing, useful discussions and advices either inside or outside of the academic areas. To name some, Akapot Tantrapiwat, Nattakan Puttarak, Vitchanetra Hongpinyo, Peiyu Tan, Xai Xie, Xingkai Bao, are those from many whom I have spent some wonderful time with during my years in Lehigh University.

Also, I would like to thank my family for their supportive roles in my degree

pursuing, my parents, my wife and my daughter.

Contents

List of Tables

List of Figures

Abstract

In the era of ubiquitous information-centric world, the amount of data that has been exchanged, stored is enormous. Everything has been connected through as a network, as we all know, the internet is the bigger than ever network. It is trending that voice and data network is indistinguishable as the handled devices are getting smarter and has powerful computation power at the level of the conventional personal computers. Though the trend is centralized scheme like client-server model is strong, distributed computing is emerging area and proven to provide better performance metrics in some senses. Peer-to-Peer application is an example of a successful of the distributed computation and communication model.

Distributed model is excellent in its redundant which gives us a fault-tolerating system or network. It can provide superior performance than working alone. Applications like cooperative communication, cooperative sensing, cooperative localization are the example of the excellence benefit over a single unit. In the future, we may discovered a lot more applications to come that exploit the distributed nature of the network.

Of course, one of the importing challenges of distributed computing is communication and managing as well as the efficiency of the operating of the algorithm/computation. In general, some algorithms may be generalized enough to be able to apply to many problems. Others, however, may not be that lucky and require specialized algorithms/schemes in order to achieve the solution. Our research falls into the latter cases, we find particular solutions that is tailored to specific applications. The applications in our interests are cooperative communications, spectrum sensing, and localization.

In the first part, we study cooperative data communication, specifically, network-coded receiver cooperation for a single-source multiple-receiver broadcasting model. Cooperative communication has been studied mostly with the senders or relays doing the cooperative relaying. Less study has been done when the receiver side cooperate to each other. We develop network coding scheme for the receiver cooperation.

Network codes based on $GF(2^q)$ random-mixing are considered complex and prone to errors. Sparse binary random-mixing is considerably simpler, but for it to be space-preserving requires the involvement of a huge number of source packets (vectors). In our case, we propose a novel strategy of an offset sparse binary random mixing at the receiver side, in which the source vectors are firstly circularly shifted, each by a different random offset, before being XORed. This simple strategy cleverly compensates for the low degree of the binary field (because of fewer number of practical receivers who in the same cooperating set) by the large dimension of the vector space. This ensures the near linear-independence of the random binary superposition while enjoying solid code structure from the well-known class of quasi-cyclic low-density parity-check codes.

We also introduce information exchanging schemes in user cooperation that would help to eliminate the need of centralized control center. By having distributed scheduling allows distributed nodes to quickly reach a rational consensus without the need to flood much of the over head or side information.

In the second part, we investigate efficient algorithm in cooperated spectrum sensing in wireless network, which is generally harsh environment caused by shadowing and/or fading. Spectrum sensing in cognitive radio considers two entities, the owner of the spectrum, both transmitters/receivers called primary users and the other set of users called secondary users, who are waiting to use the bandwidth whenever it is vacant. Each cycle consists of two stages, sense and occupy. The first step is a very important as it would help to ensure that the interference by the primary users will be minimized as well as utilizing most of the available time slot.

Cooperative sensing will improve the accuracy of the measuring step. Here, inspired by a well-known class of probabilistic inference problems, believe propagation and message passing algorithm, we model the network of the secondary users as Markov random fields and pull a group of secondary users to cooperate through distributed probabilistic inference. This results in an effective sensing and fusion strategy. The proposed framework subsumes belief propagation, as well as conventional weighted hard/soft combining (such as maximal ratio combining and equal gain combing). It can also account for the distance dependent correlation among individual sensing results by setting appropriate compatibility function. Theoretic upper and lower bounds are derived, demonstrating the significant gains made possible by effective cooperation. Extensive simulations confirm the analytical results.

Since cooperation inevitably introduces communication overhead, question arises

as how and how much cooperation should be induced to attain the low-hanging fruits without being excessive and overshadowing the gain. Based on the topology graph, we proposes a distributed tri-state probabilistic inference mechanism for cooperative sensing. Conventional decision fusion strategies pull together all the local information (e.g. yes or no for some hypothesis) in the neighborhood, irrespective of its quality. The new idea in the tri-state decision fusion is that if a cognitive radio is rather unsure (up to a threshold) about its sensing result, then instead of sending out this information (which may well be useless anyway), it might as well remain silent, staying in the third state of "erasure" to save energy (and bandwidth). Information-theoretic analysis is conducted to determine the optimal threshold that maximizes the data-rate to energy ratio. Extensive simulations are conducted which confirms the advantages of the tri-state information dissemination strategy.

We further study more into the theoretical ground of our distributed sensing framework: the belief propagation (BP) as well as a similar class of in-network computation algorithm called distributed consensus (DC). We detail the exact operations of these algorithms, unify them in the general formulation of linear dynamical systems, evaluate their convergences, and compare their behaviors. Our analysis provides useful insight into how the two algorithms arising from drastically different theoretical basis can serve a common purpose. Specifically, it is shown that belief propagation is an algorithm of "less is more" and hence favors sparse graphs, whereas distributed consensus is an algorithm of "the more the merrier" and hence favors dense graphs.

Unlike the DC algorithm where convergence to the global optimum is guaranteed irrespective of the network topology, the BP algorithm is not. We therefore propose

to trim down the graph to a minimum spanning tree (through some linear-time distributed algorithm) before running BP. This guarantees a convergence of BP to the best results in the fastest speed. Simulation results on random and structured graphs verify the analytically results.

In the third part, we study the problem of cooperative localization of a large network of nodes in integer-coordinated unit disk graphs, a simplified but useful version of general random graph. Exploiting the property that the radius r sets clear cut on the connectivity of two nodes, we propose an essential philosophy that "no connectivity is a useful information" in unit disk graphs. Exercising this philosophy, we show that the conventional network localization problem can be re-formulated to significantly reduce the search space, and that global rigidity, a necessary and sufficient condition for the existence of unique solution in general graphs, is no longer necessary.

While the problem is still NP-hard, we show that a (depth-first) tree-search algorithm with memory $O(N)$ (N is the network size) can be developed, and for practical setups, the search complexity and speed is very manageable, and in magnitudes less than the conventional problem, especially when the graph is sparse or when only very limited anchor nodes are available.

Chapter 1

Introduction

The computing and communication world is undergoing a paradigm shift from centralized to distributed architecture, and from computation within a single gigantic processing node to cooperative processing among multiple (and potential a large number of) smaller and weaker nodes. Fundamental challenge of a networked system is to pull together all the resources among the geographically-separated nodes to achieve a common task in a most efficient and trust worthy manner.

Our research goal is to study distributed signal and information processing algorithms among networked systems comprising many nodes. Examples of these networks may be ad-hoc wireless networks, cognitive radio networks, or distributed storage networks. The unifying theme of this research is to model the network systems using an appropriate graphical model and apply on it an efficient stochastic inference algorithm to achieve an information and decision consensus among the participating nodes.

Figure 1.1 represents the scope of the research area we have explored. We study

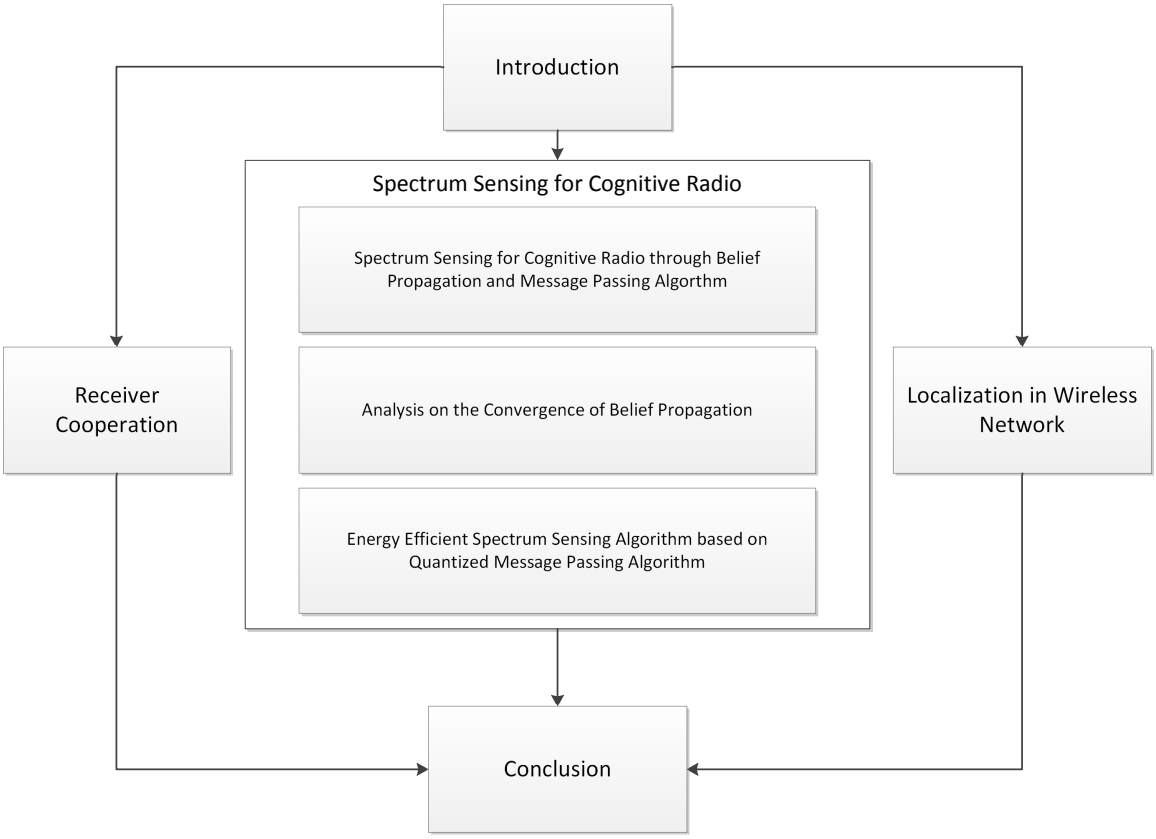


Figure 1.1: Outline

three types of applications, cooperative data communication, cooperative spectrum sensing and cooperative network localization. We give each project a brief introduction in the following sections.

1.1 Receivers Cooperative Data Communication

In wireless communication, tree-terminal communication, cooperative communication, relay communication are terms that refer to the extension of a fundamental point-to-point single-source single-destination communication. One or more relays are introduced in between the source and the destination. The task of these relays is to echo/repeat the signal from the source. Relaying may target different goals, for example, it can be used to increase communication range, increase transmission rate, or decrease the outage.

The notable gain from this relaying scheme is the diversity gain, especially in a severe or disruptive fading environments. Figure 1.2 depicts the basic model of the user cooperation. The receiver benefits from the signal from the source traveling though different paths, thus facing different channel states, which results in better diversity. In this case it is spatial diversity. Also, it does not limit to just one source/relay/or destination. It applies for many other multiple sources/relays/destinations as well. And it has been an active research topics with these combination of different settings.

When talking about spatial diversity, it is unavoidable to talk about multiple-antenna scheme like the Multiple-Input Multiple-Output (MIMO) communication or space-time coding. User cooperation is another form of diversity by design. It can be differentiated or even related to MIMO or space-time coding in the following way: if

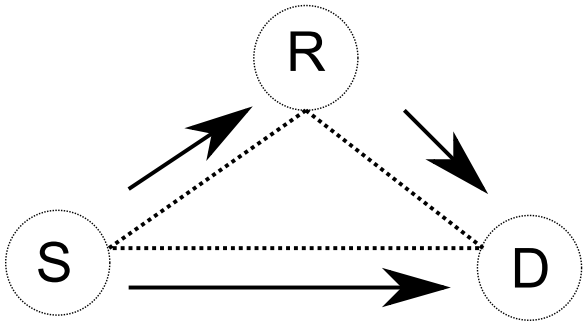


Figure 1.2: Simple 3-Terminal User Cooperation

MIMO is mostly implemented by making use of hardware, e.g. physical antenna, then user cooperation can be viewed as another form of virtual antenna, working from a higher level. This in fact, is an advantage the user cooperation has over the hardware antenna. One scenario is that implementing physical antenna may cost a lot of money and it is also limited by the size and the space between antenna. We cannot fit 2 antennas with enough spacing between them in small communication devices such as cell phones. Combining MIMO and user cooperation is not very uncommon since it is possible for the relay to have multiple antennas too.

Nevertheless, user cooperation can also be considered as one of the applications of distributed processing with the primary purpose of achieving better diversity gain. One of the well-known techniques is the user cooperation via relay networks, which is an active research for decades [1]. Several approaches such as Amplify-and-Forward (AF) and Decode-and-Forward (DF) [2] fall into this category.

AF is the scheme where the relay unit repeat and amplify the signal from the source directly in the analog form, while the DF refers to the case where the relay try to decode the source signal first before encode and forward again. AF relay is

relatively cheaper in term of hardware implementation as it is consider dumb terminal while the DF relay is a little more intelligent terminal. The advantage of DF over AF is clear when the signal from the source to the relay is good enough for the relay to decode.

Recent discovery of network coding [3] has shown significant capacity gain, which is first considered in network level, e.g. noise-free networks. The concept of network coding can be demonstrated in figure 1.3. Each of the sources A and B are transmitting to C and D. At node E, the incoming data will be XORed before transmitting to C and D. Then C and D can decode the packet from E by XORing to the direct data from their sources. This will save 1 time slot from node E. If E is not performing network coding, E would have to transmit twice. The merit of network coding in wired network is obvious, the increase of transmission rate.

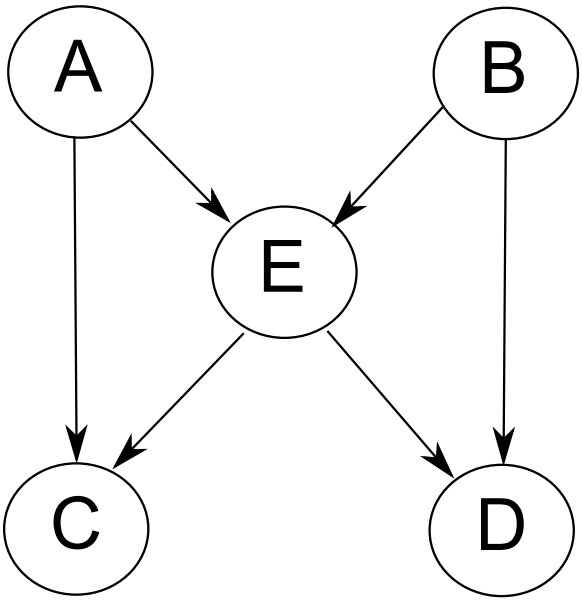


Figure 1.3: An example of Network Coding Topology

It is later found to be a useful cooperation technique in which the simple XORing used in network coding provides another form of achieving diversity gain in noisy communication channels. It is well adopted especially in a scenario where wireless nodes are formed as a network. Using the same figure (fig. 1.3), when network coding is used in user cooperation, it means that 2 pairs of sender-receiver can share the same relay, hence saving the hardware cost while preserving the diversity gain.

In static networks or networks with centralized controllers, optimized network coding scheme is proven to achieve the capacity. This model, however, does not fit well in distributed environments where topology of the networks are changing over time. So one of a good dynamic solution is random network coding. It turns out to be a practically good scheme. It can be used in both error-free network (original network coding) as well as the wireless network coding (for user cooperation).

Take a generalized network model with N number of source nodes, M number of destination nodes and R number of relay (in conventional wired network the relay is the router). We summarize the general data transmission into 3 phases.

1. The sources transmit data directly to the destination. In case of wired network, this is the standard unicast scheme. In wireless, it is the direct broadcasting from sources to the destinations, while the relays can take advantage of overheard of them. This is direct result of one of the advantage of wireless broadcasting. The packets are obviously viewed as systematic data.

2. The relays after listening to all the packets from the sources, they randomly combine the packets and re-transmit to the destinations. This part can be viewed as parity packets.

3. The packets then are collected at the destinations and decoded.

Note that in the source-destination or relay-destination phases, to avoid collision, the time-sharing scheme can be used. Many people who first learn about user cooperation with time-sharing scheme may be waste of the bandwidth but the result has shown that in wireless network, when normalized for the fair-bandwidth consumption, user cooperation out-perform the ones without cooperation. This is due to the fact that in wireless communication, channel state fluctuation is generally severe.

From the destination point of view, the packets received are formulated into a code. In general, the random-mixing network codes can be done in high order GF(2^n) as considered in [5]. An intermediate node performs linear combination in GF(2^n) on the incoming input vectors by assigning random weights and summing them as an outgoing message and passing it to the destination. At the receiver, decoding is performed to extract the original information.

In the case of error-free network, it can be also viewed as erasure channels because sometimes packets maybe dropped. Hence the code is considered as an erasure code. As a result, the received vectors must be sufficient to form a full-rank matrix, in which we can perform matrix inversion or back substitution.

If it is a noisy channel like wireless channel, decoding techniques such as LDPC codes in GF(2^n) with message passing algorithm can be used. However, high order GF(2^n) matrix inversion and message passing algorithm are considered computationally expensive. GF(2), on the other hand, allows much less complicated operations such as XOR.

Binary LDPC codes, as an example, exploit the sparseness property of graph and the binary operation, allowing efficient message passing to be performed. In

network context, this concept still holds and can be used in conjunction with binary random-mixing method, which rely on the nature of random-mixing, which states that with high probability, binary random-mixing results in a full-rank matrix and hence recoverable at the receiver using efficient algorithm, like message passing as stated before.

Recognizing the similarity and harmony of user cooperation, network coding, binary random mixing, and efficient message passing algorithm, one may view and link these building blocks together by centering on the network coding as a broader and more general set of passing information through network while enjoying the better capacity. Classical user cooperation, thus, are simply special cases of the network coding while binary random mixing is just an encoding technique performed on messages when passed through intermediate nodes. Therefore, sources, relays and destinations with appropriate encoding and decoding are the core components of the generalized cooperative networks.

1.2 Cooperative Spectrum Sensing for Cognitive Radio

In the second part, we shift our interest into the emerging area of Cognitive Radio, where network computing can be part of the solution. As many people are aware of a nature of wireless communication through the radio wave; it has only one medium, which is the radio frequencies traveling through free space. The spectrum is essentially the most scarce resource. Under the current resource allocation model. These

spectrum are divided and licensed to organizations or companies, either for commercial or military uses. Often time, the spectrum is under utilized by these companies. For example, even though a cellphone carrier reuses its frequencies through spatial dividing, the carrier may not cover some areas, or the bandwidth may not be used all the time in some rural areas where the population is sparse.

This leads to an optimistic proposal to have a dynamic spectrum sub-leasing system where the spectrum can be time-shared with other parties as long as there is no or at least controllable interference to the owner of the spectrum. The effort to design this is system is an on going research and still in its early stage. The term Cognitive Radio is meant to give an impression that the temporary leasing system should to be smart, dynamic and adaptive. The owner of the spectrum is often call *primary users* while the *secondary users* refers to the other parties who are interested in using the bandwidth.

There are quite a few possible scenarios where the Cognitive Radio can be used.

- If the transmission from the primary user is not continuous, e.g. some time slot is empty, then the secondary can use that time slot for their own purposes. This can also viewed as a time-sharing base.

- If the transmission is continue us, e.g. in TV broadcasting, it is also possible that the secondary users can use the spectrum if it is out of the communication range of the primary users (both source - destination),in this case it is the TV receiver.

To prevent the secondary from harming the primary users, there has to be a mechanism that prevent or avoid the interference. The answer is to have a spectrum

sensing before using the bandwidth. Spectrum sensing is the critical part of the whole cognitive radio.

The signal detection and estimation can be readily applied into this instance of the problem. The very common and fundamental technique is the Energy Detection. The secondary user simply set up a threshold θ. It decides to stay quite (not using the frequency) when the signal is above θ. The threshold is selected in a way that it utilize most of the available opportunity to use the bandwidth while minimize the error (interference) to the primary users.

In the noise-free environment, the two requirements can be archive simultaneously without compromising each other. However, in noisy real world characteristic of the wireless communication, noise is presence. That means some time we have a wrong decision from both sides, e.g. when we guess that the channel is available, the actual might be not and this cause the interference. On the other hand, it happens that we decide that the channel is in-use, but it is actually not, then we miss the opportunity to use the channel.

The two objectives are trade-off to each other, meaning we cannot meet those requirements at the same time. For example the threshold θ may be chosen to achieve 5% rate of the interference to the primary users. Unlike the decision in communication channel where we do not differentiate the error. In the theory of estimation, however, the probability of error consists of the probability of wasting the opportunity (to use the channel) and the probability of making unwanted interference.

Above is largely the case of single secondary user. It is interesting when a group of secondary formed into network is performing the spectrum sensing. While they are racing to get their time on the spectrum, it turns out that they can cooperatively help

each other to perform better detection accuracy. Under severe fading, cooperative sensing significantly immunize themselves from the drastic error.

The early proposed cooperative schemes focus on centralized system where a dedicated decision making unit gathers all data from the members then broadcast the result back to each individual. After that, it depends on the agreement between the secondary to decide among themselves or by the central controller whether who gets the right to use the slot.

The centralized method is good in the static type of network but it does not look appealing when the network is dynamic with constant changing all the time. The overhead of communication to the centralized unit can be overwhelmed. This is when the distributed algorithm comes in to play. It eliminate the need of the presence of the centralized computing unit. The main challenge, in this distributed environment, is how we exchange and digest the data efficiently. Conventional data flooding could be used, in such case the soft (high precision quantized value) measurements will be flooding out but it would draw a large amount of energy during the process. It is called soft combining. There are two types of combining, Maximum Ratio Combining (MRC) or Equal Gain Combining (EGC). The MRC scales the measurement by the channel state information while the EGC treats each measurements indiscriminately.

Soft information combining requires significant uses of transmitting energy. An immediate coarse-gain counter part is the hard combining, whose messages are the decision made individually. In this case it is binary hard combining. It is possible to have multi-level quantization depending on how much we want to trade the energy saving for better performance.

Either it is soft or hard combining, it does not change the fact that we have

to exchange the data somehow. A novel distributed averaging consensus algorithm has been applied to this problem. The algorithm intelligently exchanging data by just a single broadcasting in each round. In the next round, each participating node computes the output from the input heard from the neighbors. After iterations, the output value hold in each node asymptotically approaches the average value, which is equivalent to the result from MRC or EGC (with the weight over the number of node). It provides the same performance as the centralized system. The drawback, however, it may depends on the topology of the network, which causes the propagation of the data slowly and this can be more costly than the conventional algorithm. More detail will be discussed in the chapter.

We, we however, take a different approach. We propose a message passing algorithm, which is developed from a well-known class of probabilistic-inference problems. The framework is well known in machine-learning, physics communities as Bayesian network. Once the problem is formulated this way, it can be solved using message passing algorithm (or believe propagation). There are vast different kind of problems that are formulated in this way. A well-known LDPC can also be classified as a class of the problem.

Another interesting application is image inference problem where the raw image measurement data will be passed through the algorithm to find the actual images behind the noise. This is the same concept as in the LDPC decoding. The basic setting is to model all the received evidences and its relation as a graph associated with believe value. The nodes are link together by its relation or correlation. In LDPC code, each transmitted symbols are the actual value that we want to solve, the received symbols are the evidence. LDPC can be formulated into a graph where the

relation between nodes come from the code structure and encoding rules. In the case of imaging inference problem, the original pixels are what we want to solve, while the blurry pixels are the evidences.

In this imaging case, the assumption is that each pixel is correlated to the neighbors. It is modelled as Markov Random Fields and then the graph can be formulated with the connection between a pixels to the nearby pixels. Then the message passing can be used. More detail discussion can be found in the chapter.

Inspired by this idea, we can think of the state of the primary user as our unknown and each of the measurement from the secondary users as the evidences. We can then model this as Markov random fields and pull a group of secondary users to cooperate through distributed probabilistic inference algorithm. This results in an effective sensing and fusion strategy.

The proposed framework subsumes belief propagation, as well as conventional weighted hard/soft combining (such as maximal ratio combining and equal gain combing). It can also account for the distance dependent correlation among individual sensing results by setting appropriate compatibility function. Theoretic upper and lower bounds are derived, demonstrating the significant gains made possible by effective cooperation. Extensive simulations confirm the analytical results.

We further perform the analysis to make better understanding out of the proposed framework. By studying our algorithm as well as a similar class of algorithm, the average consensus, we detail the exact operations of these algorithms, unify them in the general formulation of linear dynamical systems, evaluate their convergences, and compare their behaviors.

Our analysis provides useful insight into how the algorithms arising from drastically different theoretical basis can serve a common purpose. Specifically, it is shown that belief propagation is an algorithm of "less is more" and hence favors sparse graphs, whereas distributed consensus is an algorithm of "the more the merrier" and hence favors dense graphs. Unlike the DC algorithm where convergence to the global optimum is guaranteed irrespective of the network topology, the BP algorithm is not. We therefore propose to trim down the graph to a minimum spanning tree (through some linear-time distributed algorithm) before running BP. This guarantees a convergence of BP to the best results in the fastest speed.

We also apply this with the hard-combining case, we proposes a distributed tri-state probabilistic inference mechanism for cooperative sensing. Conventional decision fusion strategies pull together all the local information (e.g. yes or no for some hypothesis) in the neighborhood, irrespective of its quality. The new idea in the tri-state decision fusion is that if a cognitive radio is rather unsure (up to a threshold) about its sensing result, then instead of sending out this information (which may well be useless anyway), it might as well remain silent, staying in the third state of "erasure" to save energy (and bandwidth). Information-theoretic analysis is conducted to determine the optimal threshold that maximizes the data-rate to energy ratio. Extensive simulations are conducted which confirms the advantages of the tri-state information dissemination strategy.

1.3 Cooperative Localization in Wireless Network

Localization and location services become norm in recent years. Global Positioning Service (GPS) system is integrated into many devices found in portable navigation units, cell phones, cars, etc. It is a new norm of technology in the recent years. The localization technique may be based on knowledge of the distances between a targeted device and three other references whose locations are known. Three reference location is sufficient and necessary condition in two-dimensional space, given that the distances are known. By solving the following non linear equations, which is equivalent to drawing a radius around each reference and the intersection between them is the location of the target node.

Let x and y be the unknown two dimension coordinate of a target device, (x^{r_i}, y^{r_i}) for $i = 1, 2, 3$ be the reference coordinates and d_i are the distances measured between target location and the references.

$$\begin{aligned}(x - x^{r_1})^2 + (y - y^{r_1})^2 &= {d_1}^2 \\ (x - x^{r_2})^2 + (y - y^{r_2})^2 &= {d_2}^2 \\ (x - x^{r_3})^2 + (y - y^{r_3})^2 &= {d_3}^2\end{aligned} \tag{1.1}$$

The distances between the reference nodes and target device is the input measurement. In practice, the distances may not be measured directly. There are several techniques and variations of the measuring methods. The well known techniques include Time Of Arrival (TOA) and Received Signal Strength (RSS). These measurements can be translated into the distance and solved by the above equation. This particular scheme is known as trilateration technique.

TOA translates time into distance using the speed of the propagated signal, for example, speed of light or speed of sound. If v is the speed of the signal and t is the propagation time, then we can translate the time into distance using the simple equation

$$d = v * t$$

RSS measurement can be translated into the distance by using the signal attenuation model. For example, one may adopt lognormal path loss model

$$h(d) = \bar{h}(d_0) + 10\alpha \log_{10}(d/d_0) + w, \quad (dB) \tag{1.2}$$

where $h(d)$ is the received signal strength at the target device and d is the distance. d_0 is a constant and w is noise.

The method describe above can be visualize as if drawing a cycle with radius $r = d$, where every point in the circle edge is the possible location. It is easy to see that drawing 3 cycles produce an intersection which is the location of the targeted device. Figure 1.4 demonstrates this scenario.

In term of accuracy, the challenge of TOA is the time synchronization between the reference stations, it is very sensitive variable that has to deal with the speed of light and a slight noise can cause huge error in the distance. RSS may suffer from the background noise and signal fading.

Time Difference Of Arrival (TDOA) is another variation from TOA where the measurement is made for the difference of the time of arrival instead the absolute

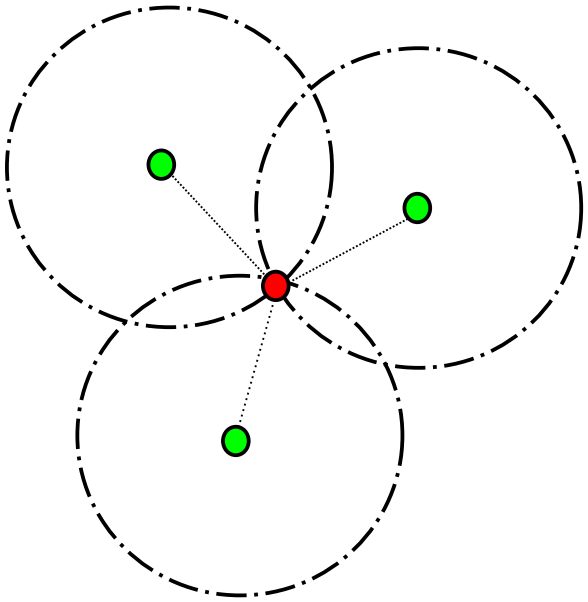


Figure 1.4: Trilateration Example

time. In TOA we got a possible location in a cycle around a reference point but TDOA procudes a hyperbolic curve of the possible locations. The intersection multiple measurements gives the location of the target.

A slightly different approach is the Angle Of Arrival (AOA) where directional angle is used to draw direct line and find the intersection between them. Note that two reference nodes are necessary and sufficient. When combine absolute distance with the angle-of-arrival technique, a straight forward way which requires only the knowledge of angle of arrival and distance from a single reference is sufficient to solve for the location. RADAR is an excellent example for this type of positioning systems. The problem of using angle information is that it tends to have larger error when the distance gets further away from the reference. This is due to slightly changes in angle results in a big gap in the far distance.

In any of the measurement techniques mentioned above, when errors and noises are accounted for, a slight error in the measurement will cause discrepancy between the measurements. In such case, estimator like the Minimum Mean Square Error can be used.

It is commonly assumed that the device looking for its location can reach at least enough reference stations. This technology, like the GPS system using the satellites as the references, is rather mature. However, in many other areas, like wireless sensor networks where fully GPS-equipped devices may not be practical. These sensing devices are often limited by the hardware cost, power or lack of the line-of-sight from the satellites.

So is it possible to locate your location if only one or two references are in the reachable range? If this is just one standalone unit, the answer is definitely no. What about cooperative and information sharing among nodes with incomplete reference sets? The answer is very possible with some challenges either in the theoretical model or in the real world. In fact some researchers have built theoretic building blocks to answer this question. Many properties or theories have been discovered. Many remain as open questions. We will leave the formal definition and detail in the chapter later on. Here we will quickly walk through some interesting development.

In [39], the simple network localization model is formulated. It simplifies the system a lot in order to make thing easier to study. The network of interest consists of many nodes forming a connected graph where its connectivity between two nodes are defined by the reachability within a unit distance. This is so called a unit-disk-graph model. Then among the nodes, there are some nodes that already acquire their locations, either by the prior fixed stationery or through the mean of GPS.

These nodes are called anchor nodes. The followings are some questions arising from this simplest settings, for example:

- Given some nodes are anchors, is it possible to resolve for the rest of the network?
- In which conditions does the network need in order to be able to find the solution (unique solution, of course)?
- Given that we have several parameters, what are the combination of the set of these parameters to produce a unique solution?
- Create theorems that give insightful information about the problem, e.g. existence of the unique solution, sufficient and necessary conditions, complexity, upper and lower bounds, etc.
- Finding algorithms that efficiently solve the problem.

Some fundamental results are shown in [39]. One interesting result is that if the graph is rigid then the unique solution exists and it requires as the same number of reference nodes as in does in the conventional GPS system. Rigidity of the graph is a key important property of the graph that would give us the prior knowledge whether the problem is solvable or not even before start to run any actual algorithm.

Rigidity of a geometric graph is analogy to the formation of connecting rigid rods together with free joints and it could tolerate any physical forces applying in any direction. The formal definition will be provided in the chapter. It is, however, as difficult as solving the problem when we want to test for this rigidity condition. Luckily, recent results show that under some conditions the rigidity can be guaranteed.

It is shown that in 2 dimensional localization. If the geometric connectivity graph representing the network has a connectivity of 6 or more for each node, then the graph can be uniquely realizable. That is 6-connectivity graph is the sufficient condition. The necessary condition up to date is when it is 3-connectivity graph. Testing this condition can be done in an efficient manner, which is a good news.

Although knowing the existence of the solution is a positive thing, solving it is a different story. It is classified as NP-hard problem. Our interest falls into this area of developing heuristic efficient algorithm to solve this problem by utilizing the underlying theories as much as possible. Many researchers formulate this problem as optimization problem (non-linear, convex, semi-definite-programming) where standard algorithms can be applied, usually in a centralized manner [42, 43].

1.4 Outline

The rest of this dissertation is organized as follows. In chapter 2, we study an adaptive network coding at the receiver side, a special class of wireless user-cooperation , which provides a performance boost via additional diversity. Although the network coding for cooperative user are general enough for any number of sources, relays, and destination, the special case of 1 source multiple receiver broadcasting case is largely ignored by other researchers.

Hence, along with this aspect, in this work, we study a special setting that consists of a single source communicating to multiple destinations in which cooperative data exchanges is allowed. We call this Receiver Cooperation. The Receiver Cooperation is particularly relevant to cases when the sender locates far away from the

receivers (satellite-ground), or when there exists a particularly weak receiver (due to fading/shadowing or antenna insensitivity). Without cooperation, the sender has to transmit and re-transmit as needed until everyone of the receivers fully received all of the packets.

In cooperative scheme, the receivers can act among themselves as virtual relays. Also the remote sender can stop re-transmitting a packet as soon as the best receiver receives all the packets, as opposed to keep transmitting until the worst receiver gets all the packets, the rest can be done via internal packets exchange through cooperative scheme. The data exchange by this way is more efficient than the ordinary broadcasting among themselves.

So we propose this adaptive network coding scheme as well as a distributed broadcast scheduling, which improves the overall performance comparing to the fixed scheduling. The analysis is also provided. It confirms that our scheme helps boosting the diversity gain.

In chapter 3 and 4, a distributed spectrum sensing technique for cognitive radio is proposed. We formulate this problem as a probabilistic inference problem based on the graph structure of secondary-users networks. Then we apply belief propagation as an algorithm to refine the decision variables. The simulation results show that within a few iterations, the algorithm converges and the performance is as good as the centralized scheme, where fusion center is presented.

We further investigate into the convergence of the message passing algorithm and we show that it reaches the optimal consensus among the participating nodes when an underlying connectivity is formed as a tree graph. For any arbitrary connected network, it can be pre-configure into a tree-like structure by employing a well-studied

distributed spanning tree algorithm.

Comparing to a similar class of the distributed consensus algorithm, the averaging consensus arising from different theoretical perspective, our algorithm gives the optimal result in finite number of iterations, depending on the network diameter d, given that the tree-like structure is used, while the averaging consensus algorithm asymptotically approaches the optimal result.

We also propose an energy efficient version of this algorithm by introducing a tri-state quantized message passing scheme. This simplifies the computation and saves the energy consumption from the communication part substantially while preserving the sensing performance as best as it could. The optimal balanced decision threshold is derived based on the capacity analysis. The simulation confirms analysis result.

In chapter 5, based on the unit-disk-graph model, we study the theory of localization, and propose a practical search algorithms for network localization that includes an implicit constraint on the unit disk graph, which significantly reduces searching space to some manageable size. We interpret the no-connectivity constraint as an important information about their locality in terms of where they cannot be. We show that an active exploitation of the forbidden region (as well as the the feasible region) can greatly change the landscape of the problem and significantly expedite the practical search algorithms.

We also show that the conventional rigidity requirement is no longer the necessary condition for the uniqueness of the solution. This means that rigid graphs become only a subset of the total set of networks where a unique solution can be found.

Chapter 2

Adaptive network coding with receiver cooperation

2.1 Introduction

User cooperation in wireless communications is considered one of the applications of distributed processing with the primary purpose of achieving better experiences in severe communication environments, e.g. in slow fading channel. One of the well-known techniques is the user cooperation via relay networks, which is an active research for decades [1]. Several approaches such as Amplify-and-Forward (AF) and Decode-and-Forward (DF) [2] fall into this category.

Recent discovery of network coding [3] has shown significant capacity gain, which is first considered in network level, e.g. noise-free networks. It is later found to be a useful cooperation technique in which the simple XORing used in network coding provides another form of achieving diversity gain in noisy communication channels. It

is well adopted especially in a scenario where wireless nodes are formed as a network.

In static networks or networks with centralized controllers, optimized network coding scheme is proven to achieve the capacity. This model, however, does not fit well in distributed environments where topology of the networks are changing over time. Dynamic approaches under the general theme of the so called random-mixing are then considered [5]. It appears to be asymptotically as good as fixed networks with an advantage of being adaptive. In general, the random-mixing network codes can be done in high order GF(2^n) as considered in [5]. An intermediate node performs linear combination in GF(2^n) on the incoming input vectors by assigning random weights and summing them as an outgoing message and passing it to the destination. At the receiver, decoding is performed to extract the original information.

In the case of error-free or erasure channels, the received vectors must be sufficient to form a full-rank matrix, in which we can perform matrix inversion. If it is experiencing a noisy channel, decoding techniques –for example, LDPC codes in GF(2^n) with message passing algorithm– can be used. However, high order GF(2^n) matrix inversion and message passing algorithm are considered computationally expensive. GF(2), on the other hand, allows much less complicated operations such as XOR. Hence, much effort have been pushed into this direction.

Binary LDPC codes, as an example, exploit the sparseness property of graph and the binary operation, allowing efficient message passing to be performed. In network context, this concept still holds and can be used in conjunction with binary random-mixing method, which rely on the nature of random-mixing, which states that with high probability, binary random-mixing results in a full-rank matrix and hence recoverable at the receiver using efficient algorithm, like message passing as

stated before.

Recognizing the similarity and harmony of user cooperation, network coding, binary random mixing, and efficient message passing algorithm, one may view and link these building blocks together by centering on the network coding as a broader and more general set of passing information through network while enjoying the better capacity. Classical user cooperations, thus, are simply special cases of the network coding while binary random mixing is just an encoding technique performed on messages when passed through intermediate nodes.

The efficient decoding algorithm is another important piece that allow the entire cooperative communication realizable. Therefore, sources, relays and destinations with appropriate encoding and decoding are the core components of the generalized cooperative networks. Hence, along with this aspect, in this work, we study a special setting that consists of a single source communicating to multiple destinations in which cooperative data exchanges is allowed. We call this Receiver Cooperation. The Receiver Cooperation is particularly relevant to cases when the sender locates far away from the receivers (satellite-ground), or when there exists a particularly weak receiver (due to fading/shadowing or antenna insensitivity), since the remote sender can stop retransmitting a packet as soon as the best receiver receives all the packets, as opposed to keep transmitting until the worst receiver gets all the packets.

This chapter is organized as follow. We first introduce Random Mixing. Our coding, mainly encoding, is developed based on this technique. The receiver cooperation adapts and employs the this technique, which allows maximum efficiency.

2.2 Random Mixing

In this section, we discuss about theoretical backgrounds behind the random mixing technique and existing contributions related to this scheme. Random coding can be viewed as if it was emerged from different research paths because of different types of problems and applications. One example is a point-to-point communication on erasure channel, where a data packet can be dropped with some error probability ϵ. The simple feedback like ARQ is a classical scheme that has been widely used. Then the feed-forward scheme like the so called rateless codes (also known as fountain codes) are invented and cleverly used to improve the efficiency of the point-to-point transmission.

The idea is that the sender keeps transmitting encoded symbols until the receiver receives a certain number of symbols, which is enough in order to decode the original data, and then response with a feedback to the sender to stop sending coded packets. Compared to the conventional scheme, this will save a lot of ARQ overhead. The key mechanism behind this success relies on the linear combination of the source packets in the high order fields allowing unlimited symbols to be generated and transmitted to the receiver. A receiver decodes the symbols by forming and solving the linear equations. The straight forward method can be done by performing the matrix inversion. Hence, linear independence of the symbols are required. It turns out that by using random mixing, the coded symbols are linearly independent with high probability.

The same concept can be readily applied for a case where symbols are limited by physical factors, for example, in data storage where the size is limited. The k data packets are decoded and store in n storage units, where $n > k$. When the errors happen, any set of k remaining storage units will be used to decode and recover

the original data. MDS codes are novel coding scheme that can handle this type of problem very well. The alternate random mixing codes also asymptotically achieve the same goal with an additional fractions of small ϵ is required.

In error free networks, optimal multi-cast transmission, where the number of transmissions is minimized, is achievable by arranging linear combining of the source packets at the relay nodes such that all receivers who receive different coded packets can decode the original data simultaneously. This scheme falls into a general class of codes known as Network Coding [3]. The right arrangements can be found by formulating the optimization over the possible paths. This is difficult in a dynamic environments; hence, the random mixing is the alternative way that can also accomplish the same goal because the nature of random mixing. Relay nodes perform packet mixing randomly and with high probability, the receivers will be able to decode after receive a certain number of packets. The study show that the overhead is not far from the optimal case [cite].

In presence of error or in noisy environments, network coding can be used as a method to provide diversity gain with an advantage of using less energy. The random version of this scheme comes automatically and enjoy the key benefit of being random, which are the flexibility, simplicity and robustness.

In particular, random matrix is the basis building block various codes are relying on its properties. Thus, in this section, we study the random matrices based on high-order finite fields $GF[q]$. Galois fields[q] or GF[q] is a mathematical technique of finite field with q elements, while $q = p^n$ when p is a prime integer and n is a positive integer. As mentioned, in the erasure correcting code, high-order fields is an important factor as well as the sparseness of the matrices which plays an important

role to construct a good and efficient matrix. Both weight and rank of a matrix are considered because of their properties to achieve the efficient decoding performance. A full-rank matrix is a key to achieve high-probability of successful decoding, while the weight of matrices indicates the sparseness of the matrices. The high-order fields matrices yield the sparseness yet keeps probability to get a full-rank matrix high.

Since the field size, sparseness and matrix's rank are interesting properties, we study the existing contributions related to these properties. T. Ho et al. [6], who investigate the benefits of coding over routing [9], give a lower bound on the success probability of a random network code, based on the form of transfer matrix determinant polynomials, while the upper bound is on the order of the inverse of *the size of the finite field*.

Presented in [7], for k data or input symbols, the number of extra (more than k) coded symbols required to give a high probability of decoding completion is referred to as the overhead. Each nonzero field element α that $P[M_{ij} = \alpha]$ is $\beta/(q-1)$, where $\beta = (\log n - c)/n$, M is a binary-random $n \times n$-matrix over $GF[q]$ and c is an arbitrary but fixed positive constant.

The results of [7] are mainly mentioned and considered in the weight and full-rank matrices. In case of the matrix is not a full-rank , the successful technique which is presented in such work is to add the linear dependencies rows that the expected number of additional rows of M is given by

$$\sum_{k=1}^{n} \binom{n}{k} \frac{1}{q^{n-k}} \left(1 - \frac{1}{q}\right)^{k} \left[1 + (q-1)\left(1 - \frac{q\rho}{q-1}\right)^{k}\right]^{n} \tag{2.1}$$

where $0 \leq \rho \leq 1$.

Moreover, to consider the probability to achieve a full rank matrix, let the average weight of a rank k matrix be the sum of the average weights of the entries, and the average weight of the ij entry be the probability that the entry is not zero [4]: $E(wtA) = \sum_{i,j} P(a_{i,j} \neq 0)$. In the case of $m \times n$-matrices with rank u, the average weight per entry is

$$\frac{(1-\frac{1}{q})(1-\frac{1}{q^u})}{(1-\frac{1}{q^m})(1-\frac{1}{q^n})} \tag{2.2}$$

By increasing the field size q, the probability of successful decoding can be made as close to one as desired [5].

Another random-coding matrices which is based on storage system is analyzed in [8]. Consider a $kr \times n$ matrix over $GF[q]$, with the r peers (users) and k-input symbols.

Theorem 1 [8]: Let D_r be the random variable denoting the dimension of the subspace spanned by all the kr code-vectors. So the probability that random code has a u-rank is given by

$$Pr(D_r = u) = \frac{\prod_{i=0}^{u-1}(q^{kr} - q^i)\prod_{i=0}^{u-1}(q^n - q^i)}{q^{nkr}\prod_{i=0}^{u-1}(q^u - q^i)} \tag{2.3}$$

While the size of k, r, and n is given by this following relation.

Theorem 2 [8]:

$$\frac{kr}{n} \geq 1 + \frac{2log_q(n)}{n} \tag{2.4}$$

There is a tradeoff between a field size and number of overhead that if the field size is larger, the additional overhead with each chunk (sub-symbol) is minimized, and the complexity of the inversion of a matrix over a larger field size at the receiver is reduced.

Theorem 3: (Efficiency of binary random-mixing) [10] Consider generating a set of parity-vectors from $|K|$ source-vectors. Each parity-vector is generated by randomly selecting a degree d according to the *robust soliton distribution*, choosing uniformly at random d distinct source-vectors (available locally), and XORing the selection. All the $|K|$ source-vectors can be successfully deduced from any $|K| + O(\sqrt{|K|}\log^2(|K|/\delta))$ parity-vectors with probability $1 - \delta$.

Theorem 4: (Rank probability of sparse binary matrix) Let $\mathbf{A} = [a_{i,j}]$ be a $(k + m) \times k$ binary random matrix ($m \geq 0$). Suppose there exists a constant δ, such that the inequalities $\delta \leq \Pr(a_{ij} = 1) \leq 1 - \delta$ hold for $i = 1, 2, ..., k + m$ and $j = 1, 2, ...k$. Then the probability of $\mathbf{A}$ having rank k approaches $\prod_{i=m+1}^{\infty}(1 - \frac{1}{2^i})$, as $k \to \infty$. Further, if every entry in the matrix has a probability p of being nonzero, where $p = (\log k - c)/k$ and c is an arbitrary but fixed positive constant, then the expected rank of $\mathbf{A}$ is $k - O(1)$.

2.3 Receiver Cooperation

Our main contribution is the development of an automatic scheduling receiver cooperation (ASRC) protocol that consists of two innovations: offset sparse binary random-mixer (OSBram) and distributed scheduler. Both strategies run in a decentralized manner without feedback, adapt automatically to the changing link state and network topology, and scale well with the increasing number of receivers.

• The limited field size (i.e. binary) makes SBram simple, but less efficient in space-preserving, unless the network code length extends very long, which requires the involvement of a huge number of source packets. OSBram cleverly compensates the low degree of the binary field by the large dimension of the vector space (the length of a packet), by performing the XOR operation on independently "offset" packets (vectors individually circularly shifted), instead of "aligned" packets (vectors in their original bit-orders).

This simple circular-shifting strategy not only preserves the simplicity of conventional SBram, and assures (near) linear-independence of superpositions like GF(2^q) random-mixing, but also finds solid structural and decoding support from the well-known class of quasi-cyclic low-density parity-check (QC-LDPC) codes.

• The distributed scheduler allows each receiver to rationally determine and adjust its order of relaying in accordance to the reception quality. Largely ignored in the cooperative literature, scheduling turns out to be rather important in some network scenarios such as receiver cooperation. We will discuss the reasons behind it, demonstrate how efficient scheduling may be achieved distributively, and provide a theoretical analysis.

The system model comprises a remote sender transmitting wirelessly to m local

receivers. Channels between receivers are relatively stable due to physical closeness, but sender-receiver channels follow a random Rayleigh fading model and vary from channel to channel and from time to time but assumed to be constant during a block of data is being transmitted. Given a transmitting source and a group of m receivers lying in the planar area with $m \times m$ reachability, we have

$$y_i = h_i x + n_i \tag{2.5}$$

$$r_i = b_i + n_i \tag{2.6}$$

where y_i is the source-destination received signal of the i^{th} node, h_i is the fading coefficient and n_i is the Additive White Gaussian Noise (AWGN), with the same SNR either of the transmitter-receiver or the receiver-receiver channels. We assume BPSK modulation scheme is used and the fading coefficient follows uniformly distributed i.i.d. random variable in $[0, 1]$. The relayed signal r_i is a broadcast signal to all $m-1$ receivers; it has the same noise level as the source-receiver but does not undergo the fading. b_i is the forwarding message from node i.

Receiver cooperation is operated in two phases: In the broadcasting phase, the sender divides a big file into a sequence of chunks (packets, vectors) and broadcasts them to all the receivers; and in the relaying phase, the receivers process and take turns to forward part or all of their observation to each other using Time Division Multiple Access (TDMA) technique. The broadcasting signal from the receivers can be represented as follow

$$b_i = \sum_{j=1}^{N} \pi(D(y_j))I(y_j)P(j) + \sum_{k=1}^{m-1} \pi(D(r_k))I(r_k)P(k) \tag{2.7}$$

where $D(.)$ is the decoding function, $I(.)$ is the indicator function with whether the decoding is successfully decoded or not.

$$I(.) = \begin{cases} 1 & D(.) \text{ decodes successfully} \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

$P(.)$ is a Bernoulli random variable whose values are drawn randomly from $(0, 1)$ with probability p and $(1 - p)$ respectively. p can be varied depending on the desire degree of the parity check. The operation is performed in $GF(2)$. The first part of 2.7 indicates the available choices of packets received directly at node i itself. The second part corresponds to the packets available from the other nodes in the relaying phase. $\pi(.)$ is the interleaving function.

For simplicity, we choose cyclic shifting function, which is considered practically good, and has been widely used in LDPC, e.g. QC-LDPC, as we have mentioned in the previous section. This scrambling method boosts the probability of full-rankness of the code if we view this from random matrix perspective. From the LDPC's point of view, this method also connects multiple short block codes to be a single long one.

Fig. 2.1 and 2.2 show the two phases of the receiver cooperation scheme operated in the first time slot (broadcasting) plus m consecutive time slot (relaying). After all, the packets received by each node will form an LDPC-like code with different

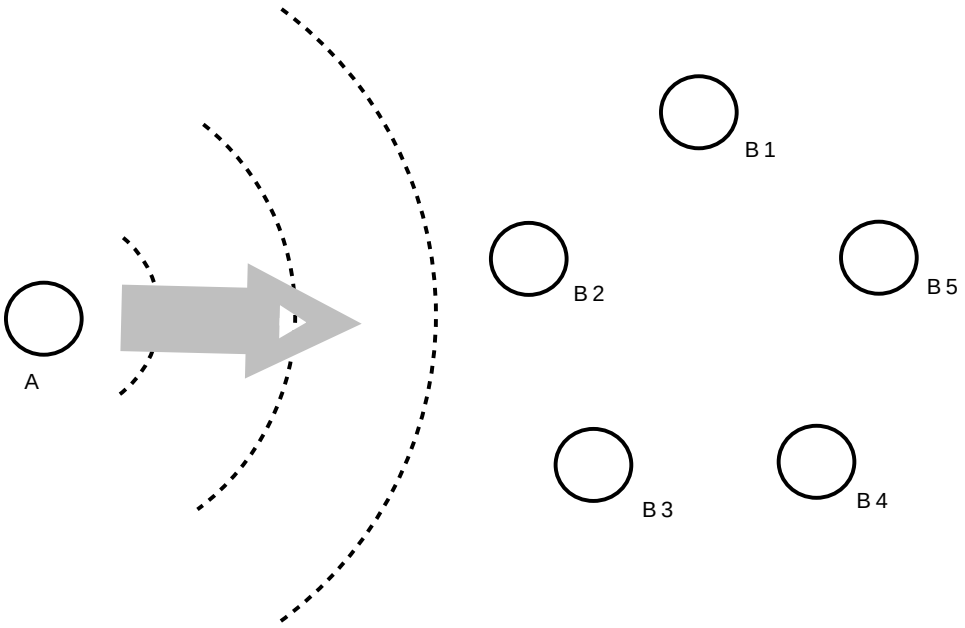


Figure 2.1: Receiver Cooperation: Broadcasting phase

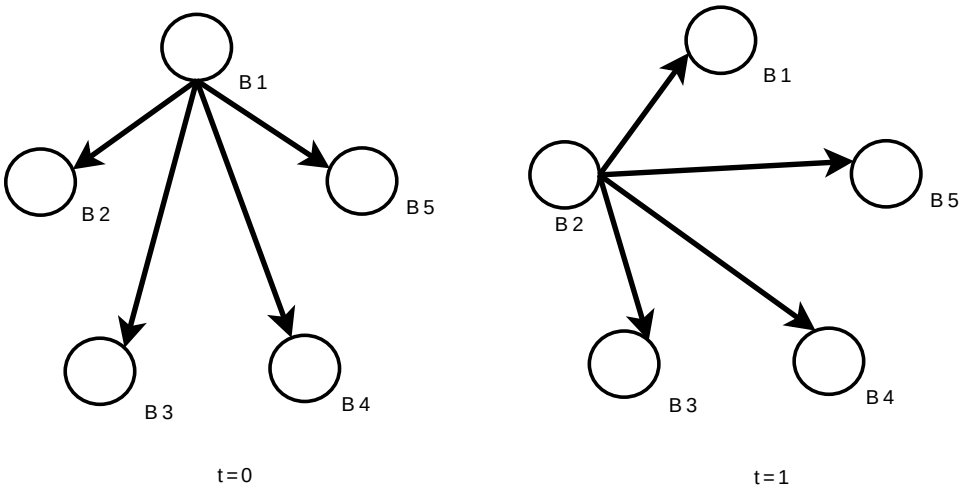


Figure 2.2: Receiver Cooperation: Relaying phase

parity check matrix on different node. Therefore, each receiver needs to know the combination of the parities in order to form the parity check matrix. A small overhead –in this case, the vector of the offset index of $\pi(.)$ and binary vector of $I(.)$– is embedded. This overhead is negligible compared to the packet size, which is relatively large. The formed parity check matrix H consists of blocks of $\pi(.)$, this time, denoted as π_{ij} taking the circular shifting identity matrix form, e.g.

$$\pi = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} \tag{2.9}$$

The H matrix falls into a class of lower triangular matrices like those of the Lower Triangular LDPC (LT-LDPC) codes. At the same time, it is also QC-LDPC codes. Fig. 2.3 depicts some examples of the matrices; fig. 2.3.A represents the original received packets that have been scrambled by the cyclic blocks, while fig 2.3.B shows the parity matrix that only make use of original received packets, and fig 2.3.C is the LT-LDPC case where broadcasting packets are in consideration when constructing the parities.

Each receiver then performs decoding using an efficient message-passing algorithm, the same one used for decoding LDPC. We will show the simulation result later. Up to this point, one may notice that in the second phase, we never mentioned about the order of relaying or even how the medium access control is done at this step. It is apparent that TDMA can be used in a round robin fashion, in the other words,

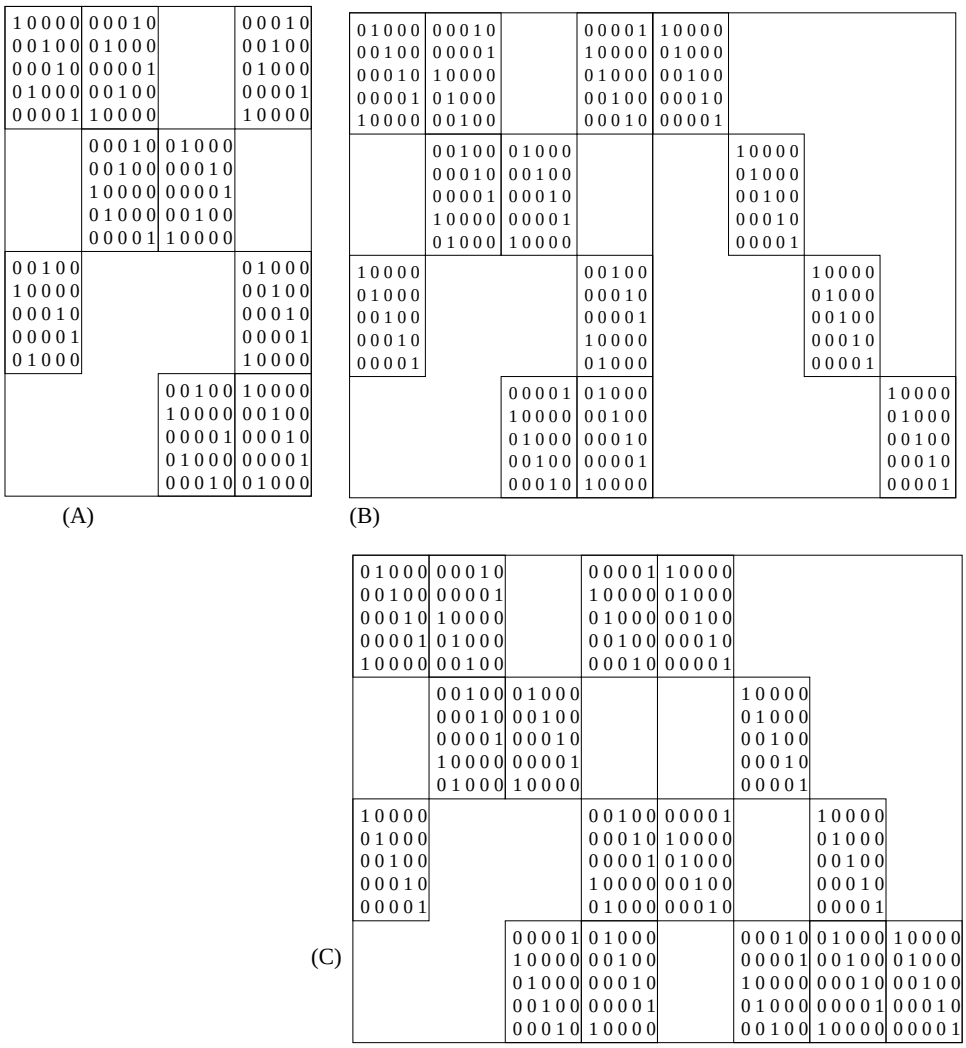


Figure 2.3: Lower Triangular Matrix

fixed order. We, however, proposed a different approach called Distributed Scheduling (DS), which not only enables distributed multiple access but also increases the system performance. The idea of DS is as follow. It first arranges the receivers based on the quality of received information (this can be the number of successfully decoded packets or the average SNR, we will discuss this later on), then using this as a broadcasting order, e.g. the best user relays first. It eventually results in performance improvement.

Intuitive explanation regarding the performance gain according to this scheme relies on the fact that each receiver only relay once and hence has to pick a combination of good packets that are available at that time; we want a receiver node to have more variety on hand and it can pick up any combination freely, which will result in a good randomness in the H matrix. However, any channel may experience an unlucky deep fade, a relaying node may end up receiving only one or zero source packet to perform effective mixing. Thus, by allowing the best node to relay first, the other nodes will receive this packet and it can be used to construct a better parity check matrix.

Another way to look at this phenomenal is that if we consider a node that does not receive anything from the source and it is about to relay first if fixed scheduling is used, then this is nothing but wasting of that time slot. Therefore, by cleverly arranging the order of relaying among receivers, the more randomness of lower triangular matrix can be obtain, which results in the better performance. As it appears, this scheduling is a critical part for receiver cooperation, which does not always naturally arise from other different scenarios of cooperation, for example, sender cooperation [11], where a group of senders broadcasting packets to a receiver in the first phase and these packets are overheard by the senders themselves, which then perform a second relay phase based on these packets. The order of relaying packets will not affect the parity

matrix of the receiver. Therefore, scheduling is not important in this case, but rather important in our scenario. It is one of the reason we emphasize this scheme here.

In general, systems whose initial broadcasting phase lacks diversity will suffer from a fixed relaying schedule. An efficient, adaptive scheduler should make the best-informed node the first to share its information, followed by the second best, and so on. But how do nodes find out, in real time, the relative reception quality of each other? Is it possible to avoid an expensive information-sharing session, and achieve the goal distributively?

A straight-forward method is to arrange an information pulling session after the broadcasting phase and before the relaying phase, such that all the nodes take turns to announce, for example, the number of source packets each receives. However, information exchange introduces additional transmission overhead and does not scale well with the size of the network. Is there a way to avoid it? Is it possible to develop a de-centralized scheduling algorithm that allows each node to compute independently without inter-node communication and still reach a rational consensus?

Our solution is simple, and naturally fits for a distributed system. Since the best-informed node should wait the shortest time to relay, we ask each node to use its reception quality, indicated by the number of packets being received, as the input to a *strictly decreasing function* g to calculate its back-off time. Pre-agreed upon by all the nodes, this strictly decreasing function ensures the consensus that a better-informed node always speaks out before a worse-informed node. To break a tie and avoid collision, a simple dithering factor is also introduced in the formula

$$\text{back}-\text{off time} = g(\text{reception quality}) + c_0 \cdot rand(\ ).$$

where c_0 is some constant. The actual form of $g(x)$ can be flexible, such as $g(x) = c_1 - c_2 x$, $g(x) = c_1 + c_2/x$, or the like, and should in general be determined by the specific application and channel conditions.

2.3.1 Performance Analysis

To evaluate the efficiency of the proposed strategy and especially the impact of adaptive scheduling, we compute the achievable information rate of two schemes, with and without adaptive scheduling(without adaptive scheduling means fixed or predefined schedule or time slot). Consider a communication system with a sender S and m receivers labeled as R_1,R_2,..,R_m. Let ϕ_i be the fading coefficient for the ith sender-receiver channel, which has probability density function (pdf) $f_{\phi,i}$ and cumulative density function (cdf) $F_{\phi,i}$. Let

$$C(\gamma) \triangleq log_2(1+\gamma)\ bit/s/Hz \tag{2.10}$$

denote the achievable information rate of a single Gaussian channel with a signal-to-noise (SNR) ratio γ.

It is well known from the information theory that in a degraded broadcast channel, the better receiver always hears all the information that the worse receiver does. The

maximal information rate for a 1-to-m receiver-cooperative system is

$$R(\phi_{sd}) = \frac{1}{2} C \left(\frac{E_b}{N_0} |\phi_{sd}|^2 \right). \tag{2.11}$$

where E_b/N_0 represents the SNR, $\phi_{sd} = \max_{i=1,2...m}(\phi_i)$ represents the best fading coefficient of all the m the sender-receiver channels in the broadcasting phase, and the factor $\frac{1}{2}$ accounts for the fact that the broadcasting phase takes only half of the time span in a cooperation round (the other half being the relaying phase). The capacity $\mathbf{C}$ of this 1-to-m system is the expectation of the information rate $R(\phi_{sd})$:

$$\mathbf{C}_m^{opt} = \int_0^\infty R(\phi) f_{\Phi_{sd}}(\phi)\, d\phi, \tag{2.12}$$

To characterize $f_{\phi_{sd}}$ in direct terms of probability distribution is difficult, but the task is much simpler in the form of cumulative distribution.

Since $\phi_{sd} = \max(\phi_1, \phi_2, \cdots, \phi_m)$, by definition of cdf,

$$\begin{aligned} F_{\phi_{sd}}(x) &= \Pr\left(\phi_{sd} \le x\right) = \Pr\left(\max_{i=1,2,\cdots,m} (\phi_i) \le x \right) \\ &= \Pr\left(\{\phi_1 \le x\} \cap \{\phi_2 \le x\} \cap \cdots \cap \{\phi_m \le x\}\right) \\ &= F_{\phi_1}(x) F_{\phi_2}(x) \cdots F_{\phi_m}(x) \\ &= F_{\phi}^m(x). \end{aligned} \tag{2.13}$$

When the fading coefficient ϕ_i of a sender-receiver channel follows a Rayleigh distribution, i.e. $f_{\phi_i}(x) = \frac{1}{2}e^{-x^2}$, we get

$$F_{\phi_i}(x) = 1 - e^{-x^2}, \tag{2.14}$$

and can subsequently compute $f_{\phi_{sd}}(x)$ from $f_{\phi_{sd}}(x) = \frac{d}{dx}F_{\phi_{sd}}(x)$.

Another way of calculating $f_{\phi_{sd}}$ this is to consider order statistic, which is defined as follow. Let $X_1, X_2, \ldots, X_n$ be i.i.d. absolutely continuously distributed random variables, and $X_{(1)}, X_{(2)}, \ldots, X_{(n)}$ be the corresponding ascendantly sorted random variables, then $X_{(k)}$ is said to be k^{th} order statistic. In our case, $\phi_{sd} = \max(\phi_1, \phi_2, \cdots, \phi_m)$ is considered as n^{th} order statistic. The useful theorem of the order statistics is that we can express the probability density function (pdf) as follow.

Let f(x) be the probability density function and F(x) be the cumulative distribution function of X_i. Then the probability density of the k^{th} statistic can be found as

$$f_{X(k)}(x) = \frac{n!}{(k-1)!(n-k)!}F(x)^{k-1}(1 - F(x))^{n-k}f(x) \tag{2.15}$$

Hence, plugging $f_\phi(x)$, $F_\phi(x)$ and $n = k = m$ into (2.15), we have

$$\begin{aligned} f_{\phi_{sd}}(x) &= \frac{m!}{(m-1)!(m-m)!}(F_\phi(x))^{m-1}(f_\phi(x)) \\ &= n(F_\phi(x))^{m-1}(f_\phi(x)) \\ &= \frac{d}{dx}F_\phi^m(x) \\ &= \frac{d}{dx}F_{\phi_{sd}}(x). \end{aligned} \tag{2.16}$$

As we can see that it leads to the same conclusion. Therefore, substitute (2.16) back into (2.12), we can evaluate the optimal capacity of the adaptive scheduling scheme.

The capacity in (2.12) represents the optimal scheduling case, where the better receivers always relay before the worse receivers. When a fixed round-robin scheduler is used, the relaying order becomes independent to the actual reception quality of the receivers: sometimes the pre-determined order happens to be efficient or (near) optimal, and other times it is inefficient or problematic.

To accurately model this phenomenon is difficult, but one may approximate the system performance by noting that in the eyes of the kth receiver, the system operates like a 1-to-k receiver-cooperative system with optimal scheduling. Thus, the capacity of the fixed round-robin scheduler can be approximated as

$$\mathbf{C}_m^{fix} = \frac{1}{m}\sum_{k=1}^{m}\mathbf{C}_k^{opt}, \tag{2.17}$$

From (2.12) and (2.17), we see that $\mathbf{C}_m^{opt}$ is always greater than $\mathbf{C}_m^{fix}$, and the difference represents the gain of adaptive scheduling. Fig. 2.4 plots the information rate of the optimal schedule and the fixed schedule for 2 and 5 cooperative receivers.

Scheduling gain is achieved even for 2 receivers, and with 5 receivers, a good schedule can increase the information rate by 22% or more (increase by 2 bit/s/Hz at a Rayleigh fading SNR of 2dB)!

In general, one can expect larger cooperative benefits for larger systems, since $\mathbf{C}_m^{opt}$ are $\mathbf{C}_m^{fix}$ are both strictly increasing functions of m. On the other hand, $\mathbf{C}_m^{opt}$ are $\mathbf{C}_m^{fix}$ will eventually saturate as m reaches some large value, at which point further enlarging the network brings but a diminishing gain. This capacity trend is demonstrated in Fig. 2.5 (for clarity, only the $\mathbf{C}_m^{opt}$ is shown).

2.3.2 Simulation Result

The simulation is conducted under the system model stated previously with m=5 receivers. The packet size is 1000 bits and the sender transmit m packets in a communication session. The receivers cooperate using the proposed cyclic sparse random-mixing method, and each receiver forwards an encoded packet once. The combined result of the broadcasting phase and the relaying phase is a (2000m; 1000m) QC LDPC (network) code of rate 1/2.

We also compare our scheme (cooperation with adaptive scheduling) against the case of no cooperation is used and also the case of cooperation with fixed scheduling. To be fair, the non-cooperative approach transmits two times using repetition code. The inter channel between receivers are kept relatively good with the channel's SNRs are randomly distributed between [γ -3db to γ+3 db]. Figure 2.6 plots the bit error rate (BER) as a function of average channel SNR γ (dB).

We consider superpositions with degree $D = 5$, i.e. a receiver randomly selects 5 of its received source/coded packets based on a constraint $E(\sum P(k)) = 5$, for all

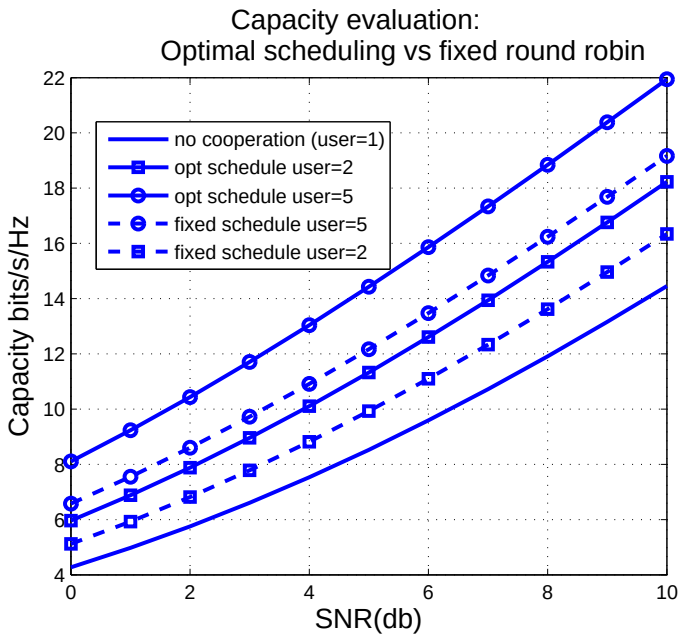


Figure 2.4: Capacity of the receiver cooperation with optimal scheduling versus fixed round robin for $m = 2, 5$.

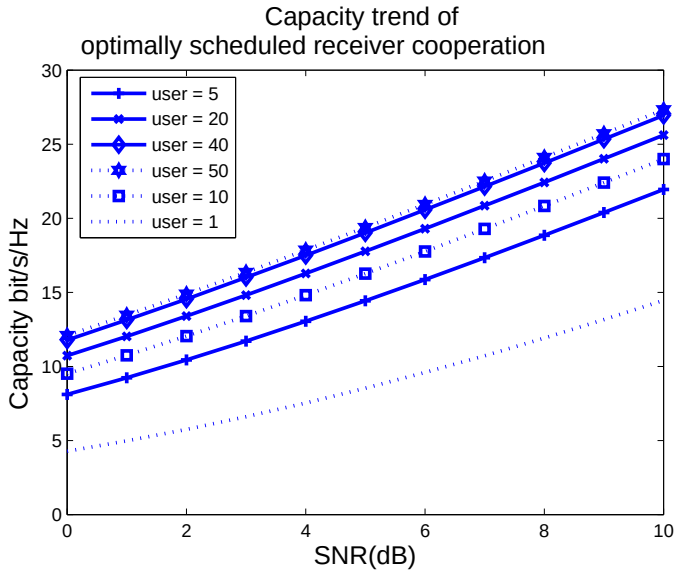


Figure 2.5: Capacity C_m^{opt} vs the number of cooperative receivers m.

k=1..2m of the original packets and the relayed packets, then circularly shifts each one and XORs them.

For comparison, we also simulated the case of direct transmission where the sender broadcasts all the source packets once[1]. At very low SNRs (BER $\geq 10^{-2}$), sender-repetition is preferred over receiver-cooperation. This is because in a harsh environment, even the best receiver can miss a one-time broadcast and hence no effective receiver-cooperation can be arranged.

As the channel conditions improve, receiver-cooperation with adaptive scheduling quickly picks up and notably outperforms all the others. However, receiver-cooperation with fixed scheduling shows only a small advantage over sender-repetition. This speaks for the importance of adaptive scheduling.

We have also simulated the case when there are $m = 50$ cooperating receivers. As is shown in Fig. 2.7, adaptive scheduling again noticeably outperforms fixed scheduling (round robin), but the margin is smaller comparing to the case of $m = 5$. This indicates that good scheduling is more critical for smaller systems, and is consistent with the analytical results in Fig. 2.4. We also evaluated the impact of different degrees D. Since $D = 10$ provides a better degree of diversity, it exhibits a better performance than $D = 5$.

However, caution should be exercised in increasing the degree, since an excessively large degree will cause the graph to be dense and cycle-abundant, which will not only significantly increase the decoding complexity, but may also severely degrade the performance.

[1]The system in this case consumes half the energy compared to the others. The comparison in the figure has compensated the 3dB difference.

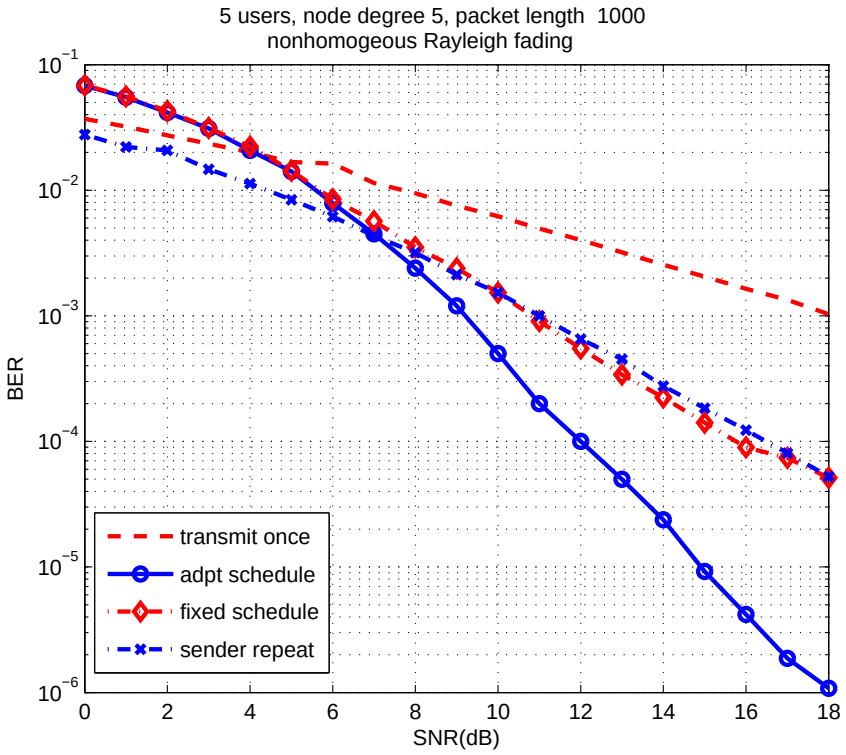


Figure 2.6: Simulation results 5-receiver systems.

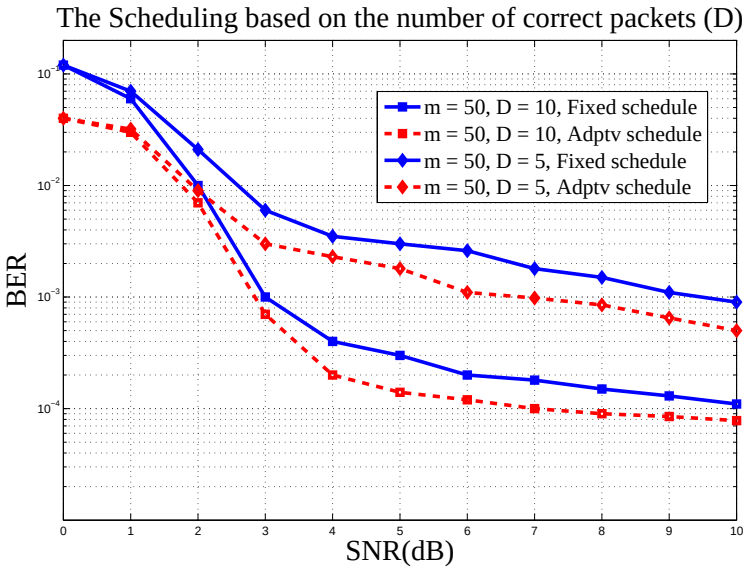


Figure 2.7: The error performance of a 50-receiver scheduling based on the number of packets D

2.4 Chapter Conclusion

User cooperation is a very useful technique for various situations. One of them is single sender to multiple receiver scenario. Distributed user cooperation with efficient strategy is the most preferable in this wireless environment. The strong background on the theory of binary random mixing and its efficiency in the implementation of field operation are the two key factors that make it very attractive for many researchers to go on this direction.

The binary random approach has been widely applied in the different forms, e.g. random network codes, fountain codes, LDPC-like codes, etc. We applied the similar concept to the receiver-cooperation case. The straight forward implementation of distributed random codes at the receivers itself is proven to be very useful; however, we further make an improvement by assigning the distributed algorithm on top of the

cooperation.

This results in significant improvement in term of the performance of the system without sacrifice any trade-off. Distributed scheduling is particularly useful in our case and make our user-cooperation scheme distinguishable from the others. In the other words, we show that in some cases, ordering or scheduling is the critical element in the user cooperation.

Chapter 3

Cooperative spectrum sensing for cognitive radio

3.1 Introduction

Cognitive radio (CR) is an effort to dynamically make use of licensed frequency band, which can be opportunistically reused by other parties (secondary users) without causing significant interference to the band's owner (primary users). The simple approach is that the secondary users sense the spectrum's usage and reuse this resource whenever it is in the idle state.

In this chapter we focus on cooperative spectrum sensing where secondary users help each other to sense the signal. Cooperation, in general, increases the sensing accuracy when the channel experiences fading especially slow fading. Several techniques have been proposed.

The basic idea is to share and gather the individual observation results among

users before making decision. Hard combining [12], is among the early techniques proposed, relies on the hard decision of each sensing node before making decision. It has been shown to be an effective technique with low inter communication overhead. Soft combining, in contrast to the hard combining, yields the fully uses of sensed information.

Several useful combination techniques, including equal gain (soft) combining, maximal ratio (soft) combining, and softened hard combination, are discussed and compared in [15]. For peer-to-peer cooperative sensing, distributed detection theory has been employed [14], and for cluster-based cooperative sensing, user selection diversity has been exploited [13]. Partial adaptive cooperation strategy that allows cognitive radios to automatically switch between cooperative and non-cooperative sensing modes in accordance to individual capacities and benefits, has also been considered [17].

Recently, a different and more sophisticated strategy using belief propagation (BP) is proposed [18]. Under the assumption that centralized computation is in use, [18] showed considerable gains enabled by BP. We, however, consider a fully distributed probabilistic inference method as another mean of facilitating cooperative sensing. A set of secondary users that cooperate to sense spectrum naturally form a *network graph*, in which a directed edge between two nodes denotes the (instantaneous) commutativity between these users.

We show that cooperative sensing can be formulated as a *system-on-graph* problem, similar in spirit to that of *code-on-graph*, and that the celebrated algorithm of belief propagation, which has found tremendous success on decoding of code graphs, can also be adapted to infer sensing decisions on network graphs.

We demonstrate, through the theoretical analysis of upper bounds and lower bounds as well as extensive simulations, that effective cooperation is particularly potential in such complex environments, and that effective cooperation can achieve magnitudes of performance better than isolated sensing.

3.2 Message Passing Algorithm

In this section we introduce the basic concepts and theories regarding various Probabilistic Inference Techniques. As an example, in previous chapter, LDPC-decoding used in the receiver-cooperation is consider one of the applications of belief propagation algorithm, a probabilistic inference technique. The well-known beleif propagation in coding community is one of the reasoning theories in the artificial intelligence (AI) area. It can be viewed as a variant of Bayesian's network.

The chapter includes a handful of theories/techniques as well as concepts, such as belief propagation, generalized belief propagation, Bays' network and Dempster-Shafer theory, Markov Networks. They are all related and shared or overlapped into some extent. In this chapter will introduce the basic background and their relations. We then demonstrate the application of them in some specific applications such as in spectrum sensing, especially in the distributed environments. Some of the reasons these techniques are so popular and suitable for distributed environments are because they can be represented graphically, thus some times called graphical models, and in some cases, they can operate iteratively.

3.2.1 Baysian Network

3.3 System Model

For per-node basic spectrum sensing, we consider power detection, which is simple, as well as least restricted and practical as verified by experiments [16]. Our model accounts for not only additive noise and random fading, but also severe shadowing, a factor that was previously largely ignored, but recently shown to be detrimental to many wireless communication and sensing functions.

In the sequel, when dealing with a lognormal random variable x, we use μ_x and σ_x^2 to denote its mean and variance, and use $\dot{\mu}_x$ and $\dot{\sigma}_x^2$ to denote the mean and variance of a Gaussian variable whose exponent is lognormal x.

Consider a set of n secondary users scattered near the guard band of a primary user, forming a topology as shown in Figure 3.1.

We consider a complex and practical wireless channel model that is dominated by large-scale shadowing (and random fading) along with log-distance path loss. Let d_i be the distance between the PU and the ith SU. The instantaneous channel gain $h(d_i)$, can be modeled as the combined effect of an average path loss and a lognormal shadowing/fading effect, which, when measured in decibel (dB), becomes normal distributed:

$$h(d_i) = \bar{h}(d_0) + 10\alpha \log_{10}(d_i/d_0) + w, \quad (dB) \tag{3.1}$$

where $\bar{h}(\cdot)$ stands for the average path loss, d_0 is some close-in reference distance, α

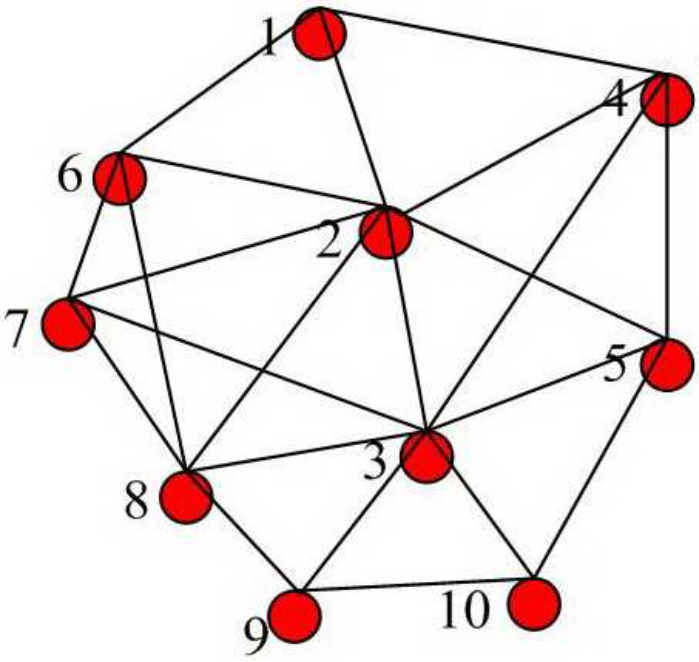


Figure 3.1: Topology of a loopy network.

is the path loss exponent that generally takes value between 2 to 4, and w (in dB) is a Gaussian random variable with zero mean and variance σ_w^2, subsuming the factors of shadowing and fading. The sensed signal y_i at the ith SU is given by

$$y_i = h_i x_i + z_i, \tag{3.2}$$

where $h_i = h(d_i)$ is lognormal,and z_i is the additive white Gaussian noise with zero-mean and variance σ_z^2.

Spectrum sensing involves determining one of following two hypotheses

$$H_0 : \text{The target spectrum is currently idle.} \tag{3.3}$$

$$H_1 : \text{The target spectrum is currently active.} \tag{3.4}$$

The decision may be made independently by each SU. For better accuracy, cooperation can arranged to allow a group of co-located SUs leverage sensing results of each other. Effective decision fusion can be performed on *soft* information, namely, the likelihood ratio test $p(H_0|y)$ versus $p(H_1|y)$, where y is a random variable or a random vector denoting the sensed signal power by the SU.

The decision of whether the target spectrum is idle for re-use at the ith SU can be made via δ-accuracy detection which guarantees that the probability of *false opportunity* (i.e. interfering with the PU), $p(H_0|H_1)$, does not exceed δ. Let θ be the critical threshold for interfering with the PU. We decide this piece of spectrum is re-usable if $\Pr(H_1|Y = y) < \theta$, and unavailable otherwise. At any desire value of δ, the threshold θ can be calculated via the following relation:

$$\delta \geq p(H_0|H_1) = \int_{-\infty}^{\theta} p(y|H_1)dy, \tag{3.5}$$

This method is known as the Neyman-Pearson method [20]. Another important assessment for sensing and detection is the probability of *missed opportunity*, $p(H_1|H_0)$. It is desirable to have both false opportunity and missed opportunity under control, but since they are two competing factors, compromise is usually made to best satisfy the application requirement.

3.4 Distributed Spectrum Sensing via Belief Propagation

Belief Propagation is a general class of message passing and updating algorithm. As its name may have revealed, this algorithm works under the following principles; a system consisting of multiple entities/units form a computing group, an entity passes an amount of its own belief (may be normalized between [0,1]) from one to the others. This values then will be combined in some way to update its local belief.

The process repeat iteratively and help refining the belief values and hoping to approach the actual value. The Belief Propagation (BP) is often represented in graphical forms, where vertices (variable nodes) represent values of interest while edges represent messages from variable nodes that influent to the others nodes. Factor graphs, Markov's random fields and Bayesian networks are some variations of the graphical model adapted to different problems [27].

BP is often used to solve inference problems, e.g. given an observed vector of random variable $Y = (Y_1, Y_2, ..., Y_n)$, we want to infer to another random vector $X = (X_1, X_2, ..., X_n)$. In many cases, the $P(X|Y)$ are unknown or difficult to compute but the local $P(X_i|Y_i)$ is available. BP takes these local values and computes for the whole vector $P(X|Y)$, then determines X.

Now, consider cooperative sensing, Y refers to a vector of the observed power on SUs and X represents the spectrum's occupancy, e.g. $X_i = 1$ means the spectrum is in use, and idle otherwise. To apply the BP to this problem, consider modeling the set of secondary users as an undirected graph, where each SU i becomes a vertex $i \in V$ and the set of edges E denote one-hop reachability.

Consider a practical topology such as that shown in Fig. 3.1. We can convert a loopy graph into equivalent pairwise MRF such that each node need only interact with its neighbors. The Hammersley-Clifford theorem [27] states that a joint distribution can be represented by a pairwise MRF of the following form:

$$p(\mathbf{X_i}) \propto \prod_{(i,j)\in E} \psi_{ij}(X_i, X_j) \prod_{j\in V} \phi_i(X_i), \tag{3.6}$$

where $\phi_i(X_i)$ is the *local function.* and $\psi_{i,j}(X_i, X_j)$ is the *compatibility function.*

In our context, we assign the likelihood values $p(Y = y|H_0)$ and $p(Y = y|H_1)$ as the local function $\phi_i(X_i)$, which can be viewed as Bernoulli distribution whose values are the likelihood of the spectrum's hypothesis. The compatibility function will be defined and explained more in the next section.

The message from node i to j at the tth iteration generally takes a probabilistic form, and denotes the current *belief* that node i has about X_j, which is computed by gathering node i's own observation and all the messages from the rest of the network in the last round:

$$m_{ij}^{(t)}(X_j) = \int \psi_{ij}(X_i, X_j)\phi(X_i) \prod_{k\in N(i)\backslash j} m_{ki}^{(t-1)}(X_i) dX_i \tag{3.7}$$

where $N(i)\backslash j$ means the neighbors of node i except for node j. The total *belief* node i has about its own variable X_i is updated as

$$b_i^{(t)}(X_i) \propto \phi_i(X_i) \prod_{k \in N(i)} m_{ki}^{(t)}(X_i). \tag{3.8}$$

The belief propagation algorithm can be applied on any network topology including loopy graphs [27]. The message or *belief* gets updated following the rules in (3.7) and (3.8) until it converges to a (local) optimal solution or when a maximum number of iterations are reached.

An important element of the BP algorithm is the *compatibility function* $\psi_{ij}(X_i, X_j)$, which directly influences the system performance and computational complexity. In general, the compatibility function is reciprocal (i.e. $\psi_{ij}(X_i, X_j) = \psi_{ji}(X_j, X_i)$) and takes a non-negative value between 0 and 1, where 1 indicates a strong influence on X_j from X_i and 0 indicates no direct influence. Specifically, the compatibility function here indicates how much one node should weigh or value the sensing result of another.

Since spectrum utilization is geolocation dependent, it is reasonable to set $\psi_{ij}(X_i, X_j)$ as some deceasing and bounded function of the distance between node i and node j. This function may take either continuous values (i.e. weight is quite sensitive to distance) or discrete values with different levels of granularity.

Without going into the burden of conducting extensive field tests to derive an exact compatibility expression as a function of distance, here we adopt a simple but sensible binary function: A pair of nodes that are one-hop reachable are assumed to have direct impact on each other, and not otherwise. Also, consider $\psi_{ij}(X_i = 1, X_j = 0)$, this reflects the compatibility or how likely that node i observes 1 while its neighbor

observes 0.

The evaluation of $\psi_{ij}(X_i = 1, X_j = 0)$ does not necessary become 0 (or totally impossible), it can be any value from 0 to 1 depending on how we want to model the reality. However, with the prior knowledge that it's more likely that if a node observes 1, so does its neighbor. We want to include this into the compatibility as well. Thus,

$$\psi_{ij} = \psi_{ji} = \begin{cases} 1, & i \text{ and } j \text{ have a direct link and } X_i = X_j \\ 0. & \text{otherwise} \end{cases} \tag{3.9}$$

3.4.1 Distributed Consensus Averaging for Spectrum Sensing

Distributed Consensus Averaging is an distributed algorithm associated with a undirected graph model. Each vertex is associated with an input value x_i. At each iteration (t), a computation is performed on each node taking the following form

$$x_i^{(t)} = w_{ii} x_i^{(t-1)} + \sum_{j \in N_i} w_{ij} x_j^{(t-1)} \tag{3.10}$$

N_i is a set of x_i's neighbors. As t goes to ∞, for some choice of weights w_{ij}, x_i converges to a desire value; in this instance, the average value is considered. The above equation can be written in a matrix form

$$x^{(t)} = Wx^{(t-1)} \tag{3.11}$$

$$= W(Wx^{(t-2)}) \tag{3.12}$$

$$= W^t x^0 \tag{3.13}$$

The element w_{ij} is constrained to 0, enforcing the no-direct-connectivity between the two nodes i, j. The weights must be chosen in such a way that

$$\lim_{t->\infty} W^t = \frac{11^T}{n} \tag{3.14}$$

One of the well known choice of weights is

$$W = I - \alpha AA^T \tag{3.15}$$

$$= I - \alpha(D - N) \tag{3.16}$$

$$= I - \alpha L \tag{3.17}$$

where A is the node-edge incident matrix of the graph, and the product between A and A^T is equal to the node degree matrix D –containing the node degree in the diagonal– and the adjacency matrix N representing node to node connectivity, while L is known as Laplacian Matrix.

$$\begin{bmatrix} d_0 & 0 & -1 & .. \\ 0 & d_1 & 0 & .. \\ -1 & 0 & .. & .. \\ .. & .. & .. & d_k \end{bmatrix}$$

where

$$l_{ij} = \begin{cases} d_i & , i = j \\ -1 & , \text{i and j are connected by an edge} \\ 0 & , \text{otherwise} \end{cases} \tag{3.18}$$

It is shown that by picking $\alpha = \left[0, \frac{1}{d_{max}}\right]$ will guarantee the convergence [21]. d_{max} is the maximum edge degree in the graph. Note that the optimal constant weight is $\alpha = \frac{2}{\lambda_N(L)+\lambda_2(L)}$ where $\lambda_N(L)$ is the largest eigenvalue $\lambda_2(L)$ is the second smallest eigenvalue.

The value of α does not need to be constant applied globally throughout the network, it is possible to vary the value based on the local knowledge $\alpha = \frac{1}{max[d_i,d_j]}$, the maximum degree between nodes associated on both ends of an edge.

When breaking down into each individual computing node, the equation 3.10 becomes

$$x_i^{(t)} = (1 - d_i)x_i^{(t-1)} + \alpha \sum_{j \in N_i} x_j^{(t-1)} \tag{3.19}$$

$$x_i^{(t)} = x_i^{(t-1)} + \alpha \sum_{j \in N_i} (x_j^{(t-1)} - x_i^{(t-1)}) \tag{3.20}$$

The equation can be interpreted as at each iteration the computing node combines its value with the differences between itself and the other connected surrounding nodes. More intuitively, each node is trying to decrease the potential differences between itself and the neighbors, doing so by distributing the differences among them. If the difference is greater than zero, that means it has to give the value to the other party. If it is negative, then it receives the incoming value. It can be easily seen that the mass conservative rule holds in this case. The constant α is needed for the convergence of the algorithm as stated earlier.

One can think of this as the resistance or conductivity of the circuitry if its value is 1, it's certain that the value of each node will be oscillating indefinitely. The best analogy for this is to model nodes as water containers filled with different water level and connected with pipes (edges), once all valves are open, the water flows and stay at the same level.

The distributed consensus algorithm have been applied into the context of distributed cooperative spectrum sensing [22, 23]. The input vector x can be the log-likelihood ratio observed from nodes, it can be other sensed data such as energy or power obtained from various measurement techniques. In this paper, the log-likelihood is assumed since it's compatible with the belief propagation.

There is a further extension to the previous DC algorithm. By introducing memory or more delayed variables to the equation, the extension is called high-order DC

algorithm and the previous DC algorithm can be viewed as the first order. The generalized form of the higher order is as follow:

$$x_i^{(t)} = x_i^{(t-1)} + \sum_{m=0}^{M} (-\gamma)^m \Delta x_i^{(t-m)} \tag{3.21}$$

where

$$\Delta x_i^{(t-m)} = \alpha \sum_{j \in N_i} (x_j^{(t-m)} - x_i^{(t-m)}) \tag{3.22}$$

3.4.2 Other Degenerated Cooperative Strategies

Majority Vote: The simplest decision fusion is hard combining with majority vote. Each user compares its (time-averaged) sensing results with a critical threshold θ to make a binary (hard) decision. In a centralized process, all the hard decisions are routed, possibly through multiple hops, to a processing center (which may or may not be one of the cognitive radios).

The centralized decision is made by the majority vote rule and distributed as a consensus to all the users in the systems. In a distributed process, each node collects its immediate neighbors' hard decision and combine them with its own to perform majority vote. Hence, each user is exposed to a different sample space, and the well-connected nodes at the center tend to be more informative and make better decisions than, for examples, nodes at the edge of the network.

Equal-Gain Combining: Soft combining tends to outperform hard combining. Like

hard combining, the process can also take a centralized form or a distributed form. Users exchange their raw sensing results or their sufficient statistics (e.g. the time-averaged version), and the Neyman-Pearson decision is made based on the combination of the soft information at hand.

Maximum-Ratio Combining: Maximum ratio combining is very similar to equal-gain combining, except that users now exchange the *normalized* sensing results or their sufficient statistics, where normalization is performed by dividing the sensing result by the instantaneous channel gain. This is equivalent to exchanging the log-likelihood ratio (LLR).

Comments: These three conventional cooperative strategies, when operated in a distributed manner, are like single-iteration belief propagation with messages taking on different forms. The decision accuracy differs from SU to SU, and is highly dependent on the neighborhood connectivity and sensing results. In comparison, their centralized realization enforces a uniform decision on every node. For small networks such as what we discuss in this paper, all the SUs are expected to experience the same influence from the PU, and centralized decision will therefore generate better sensing accuracy than distributed processing.

We caution, however, that the routing and communication overhead caused by gathering observations and distributing decision makes the centralized algorithm not only scale poorly with the size of the network, but the "uniform" decision may become inaccurate for some parts of the network. This is because for large networks, it is possible that the SUs at the near end must be alert of the activities by the PU, while the SUs at the far end can be ignorant.

3.5 Degenerated Cooperative Strategies

For comparison reason, we summarize the basic ideas of the following strategies and reestablish their results.

Majority Vote: The simplest decision fusion is hard combining with majority vote. Each user compares its (time-averaged) sensing results with a critical threshold θ to make a binary (hard) decision. In a centralized process, all the hard decisions are routed, possibly through multiple hops, to a processing center (which may or may not be one of the cognitive radios). The centralized decision is made by the majority vote rule and distributed as a consensus to all the users in the systems.

In a distributed process, each node collects its immediate neighbors' hard decision and combine them with its own to perform majority vote. Hence, each user is exposed to a different sample space, and the well-connected nodes at the center tend to be more informative and make better decisions than, for examples, nodes at the edge of the network.

Equal-Gain Combining: Soft combining tends to outperform hard combining. Like hard combining, the process can also take a centralized form or a distributed form. Users exchange their raw sensing results or their sufficient statistics (e.g. the time-averaged version), and the Neyman-Pearson decision is made based on the combination of the soft information at hand.

Maximum-Ratio Combining: Maximum ratio combining is very similar to equal-gain combining, except that users now exchange the *normalized* sensing results or their sufficient statistics, where normalization is performed by dividing the sensing result by the instantaneous channel gain. This is equivalent to exchanging the log-likelihood ratio (LLR).

Comments: These three conventional cooperative strategies, when operated in a distributed manner, are like single-iteration belief propagation with messages taking on different forms. The decision accuracy differs from SU to SU, and is highly dependent on the neighborhood connectivity and sensing results. In comparison, their centralized realization enforces a uniform decision on every node. For small networks such as what we discuss in this paper, all the SUs are expected to experience the same influence from the PU, and centralized decision will therefore generate better sensing accuracy than distributed processing.

We caution, however, that the routing and communication overhead caused by gathering observations and distributing decision makes the centralized algorithm not only scale poorly with the size of the network, but the "uniform" decision may become inaccurate for some parts of the network. This is because for large networks, it is possible that the SUs at the near end must be alert of the activities by the PU, while the SUs at the far end can be ignorant.

3.6 System Analysis

It is well recognized that lognormal shadowing, being geo-dependent, large-scale and slow varying, has a significant impact on wireless communication and sensing capabilities. Below we study the effect of shadowing. We use MSE and the probability of false opportunity as metrics.

3.6.1 Mean Square Error

Mean Square Error (MSE) is a commonly used as a tool for evaluating estimation performance. Let y_i be the communication power (of the primary user) sensed by the ith SU. From (4.2), y_i is the sum of a lognormal random variable $h_i x$ (treating x as a constant) and a Gaussian random variable $z_i \sim \mathcal{N}(0, \sigma_{zi}^2)$. The lognormal variable $h_i x$ has parameters $\dot{\mu}_{h_i x} = \ln(x) + \dot{\mu}_{h_i}$ and $\dot{\sigma}_{h_i x}^2 = \dot{\sigma}_{h_i}^2$, and without loss of generality, we may assume $\dot{\mu}_{h_i} = 0$, such that $\dot{\mu}_{h_i x} = \ln(x)$.

No Cooperation: Without cooperation, each SU i makes estimation based on its own observation y_i. For a lognormal random variable y, since its mean value μ_y relates to the parameters $\dot{\sigma}_y^2$ and $\dot{\mu}_y$ by

$$\mu_y = e^{\dot{\mu}_y + \dot{\sigma}_y^2/2}, \tag{3.23}$$

it can be shown that the MSE for a maximum likelihood detection is given by

$$\begin{aligned} \mathrm{MSE(y_i)} &= E[(E[y_i] - x)^2] = \sigma_{h_i x}^2 + \sigma_z^2 \\ &= (e^{\dot{\sigma}_{hi}^2} - 1) e^{2\ln x + \dot{\sigma}_{hi}^2} + \sigma_z^2. \end{aligned} \tag{3.24}$$

Full cooperation: Consider centralized soft combining. Soft sensing results from all the SUs will be gathered to form a new random variable:

$$B = \frac{1}{n}\sum_{i=1}^{n} y_i = \frac{\sum_{i=1}^{n} h_i x}{n} + z \triangleq \frac{A}{n} + z, \tag{3.25}$$

based on which a decision will be made. Since the sum of independent lognormal random variables is also lognormal (for a small number of terms), the random variable A can be approximated as lognormal with parameters:

$$\dot{\sigma}_A^2 = \log\left[\frac{\sum_{i=1}^{n} e^{2\ln(x)+\dot{\sigma}_{hi}^2}(e^{\dot{\sigma}_{hi}^2}-1)}{(\sum e^{\ln(x)+\dot{\sigma}_{hi}^2/2})^2} + 1\right], \tag{3.26}$$

$$\dot{\mu}_A = \log\left[\sum_{i=1}^{n} e^{\ln(x)+\dot{\sigma}_{hi}^2/2}\right] - \frac{\dot{\sigma}_A^2}{2}. \tag{3.27}$$

The resultant MSE (for centralized cooperation) becomes:

$$\mathrm{MSE(B)} = \frac{1}{\mathrm{n}^2}\left[(\mathrm{e}^{\dot{\sigma}_\mathrm{A}^2}-1)\mathrm{e}^{2\dot{\mu}_\mathrm{A}+\dot{\sigma}_\mathrm{A}^2}\right] + \sigma_\mathrm{z}^2. \tag{3.28}$$

In (3.24) and (3.28), we have included noise uncertainty σ_z^2. If we assume that each SU can perform adequate time-average to cancel noise z before making individual or centralized decisions, then we can set $\sigma_z^2 = 0$ to get the ideal lower bound for MSE.

Figure 3.2 plots the MSE bounds with different levels of noise uncertainty for both non-cooperative detection and centralized detection. The performance trend is plotted as a function of the number of users n. It is clear that noise uncertainty negatively affects the performance, but effective cooperation with a sufficient number

of users (such as $n \geq 10$) can drastically improve the situation.

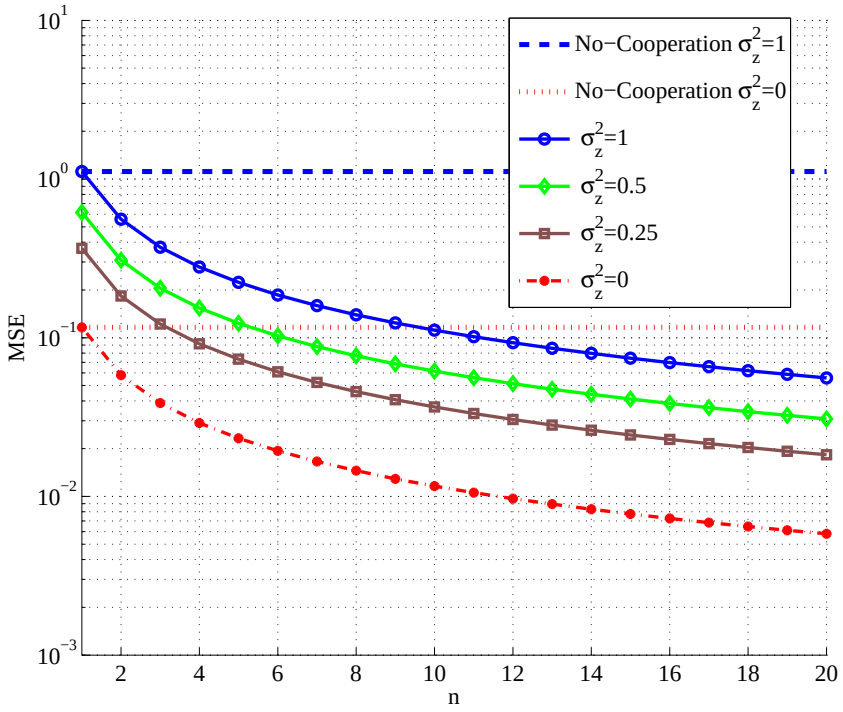


Figure 3.2: MSE(dB) as the number of cooperative users increases.

3.6.2 Probability of False Opportunity

It is of paramount importance in CR that SUs must not interfere with PU. We now evaluate the probability of false opportunity for different cooperative strategies.

No Cooperation: Using the Neyman-Pearson method, the power estimate of the individual sensing result, y_i, will be compared to an appropriate threshold, and the left tailing probability (when PU is active) denotes the probability of false opportunity and the right tailing probability (when PU is idle) denotes the probability of missed opportunity. When the system is free from noise uncertainty, y_i simply follows a

lognormal distribution.

In general, since $y_i = h_i x + z_i$, and z_i is independent of $h_i x$, the probability density function (pdf) of y_i, denoted by f_y, is the convolution of a lognormal random variable $h_i x$ and a Gaussian random variable z_i. While we do not have a closed-form expression for f_y, numerical evaluation can be carried out to compute the probability of false opportunities (assuming PU is active, i.e. x is some positive value):

$$\text{no-coop: } p(H_0|H_1) = p(y \leq \theta) = \int_{-\infty}^{\theta} f_y(t)dt, \tag{3.29}$$

where θ is the (predetermined) critical threshold.

Soft-Combining: The case of centralized soft-combining follows much the same train of thought as in the previous discussion. Instead of evaluating the distribution of y_i, here we must evaluate the distribution of B in (3.25). Since the sum of independent lognormal is still lognormal and the sum of independent Gaussian is still Gaussian, the pdf of B will again be the convolution of a lognormal and a Gaussian distribution in general, and becomes lognormal when void of noise uncertainty. The probability of false opportunity can be evaluated numerically through

$$\text{s-comb: } p(H_0|H_1) = p(B \leq \theta) = \int_{-\infty}^{\theta} f_B(t)dt, \tag{3.30}$$

Hard-Combining: The analysis of hard-combining is somewhat more involved. The ith SU makes its only binary decision by comparing y_i against a threshold θ, and a joint decision will be made based on the majority vote rule. Hence, the joint decision

will be H_0 if and only if at least half the SUs has individually decided H_0. To analytically quantify this probability, we resort to the Mathematical tool of order statistics.

Consider n i.i.d. random variables defined over the region $[a, b]$ with pdf f_y and cumulative density function (cdf) F_y. Suppose we order their realizations from small to large, and let $y_1 \leq y_2 \leq \cdots \leq y_n$ denote the order statistics of these random samples. It is easy to express the pdf of any order statistic, y_k, in terms of f_y and F_y by the following integral:

$$f_{y_k}(t_k) = \int_a^{t_k} \int_a^{t_{k-1}} \cdots \int_a^{t_2} \int_{t_k}^{b} \int_{t_{k+1}}^{b} \cdots \int_{t_{n-1}}^{b} n! \\ f_y(t_1) f_y(t_2) \cdots f_y(t_n) dt_n \cdots dt_{t+1} dt_1 \cdots dt_{k-1}.$$

After simplification, we arrive at

$$f_{y_k}(t) = \begin{cases} \frac{n!}{(k-1)!(n-k)!} [F_y(t)]^{k-1} [1 - F_y(t)]^{n-k} f_y(t), \\ \qquad\qquad \text{when} \quad a \leq t \leq b; \\ 0, \qquad\qquad \text{otherwise} \end{cases}$$

Hence, the fact that at least $\lceil n/2 \rceil$ SUs have determined a spectrum idle translates to the fact the that $\lceil n/2 \rceil$th order statistic, $y_{\lceil n/2 \rceil}$, has made a decision of H_0, i.e. $y_{\lceil n/2 \rceil} \leq \theta$. Thus the probability of false opportunity is

$$\text{h-comb: } p(H_0|H_1) = p(y_{\lceil n/2 \rceil} < \theta) = \int_{\infty}^{\theta} f_{y_{\lceil n/2 \rceil}}(t)dt.$$

Fig. 3.3 plots the probability of false opportunity as a function of the number of users n. Three strategies, no cooperation, hard-combing (majority vote), and soft-combining, are evaluated. In all the cases, we set x=1, $\theta = 0.5$, and $\sigma_{h_i}^2 = 1$. The set of solid lines and dashed lines represent the noisy case with $\sigma_z^2 = 1$, and the noise-free case with $\sigma_z^2 = 0$. The huge gain achievable by soft- and hard-combining clearly indicates the benefits of cooperation. It is interesting to note that soft-combing works effectively regardless of the noise level, whereas hard-decision combining works better in clear-noise environments than otherwise.

3.6.3 On the Convergence of Belief Propagation in Cognitive Radio Sensing

Consider the eqn. (3.7), as restated below

$$m_{ij}^{(t)}(X_j) = \int \psi_{ij}(X_i, X_j)\phi(X_i) \prod_{k \in N(i) \setminus j} m_{ki}^{(t-1)}(X_i)dX_i$$

We recall that X is a Bernoulli distribution whose values are the likelihood of the spectrum's hypothesis with two realizations $X_j = 0$ and $X_j = 1$. Then eqn. (3.7) can be simplified as

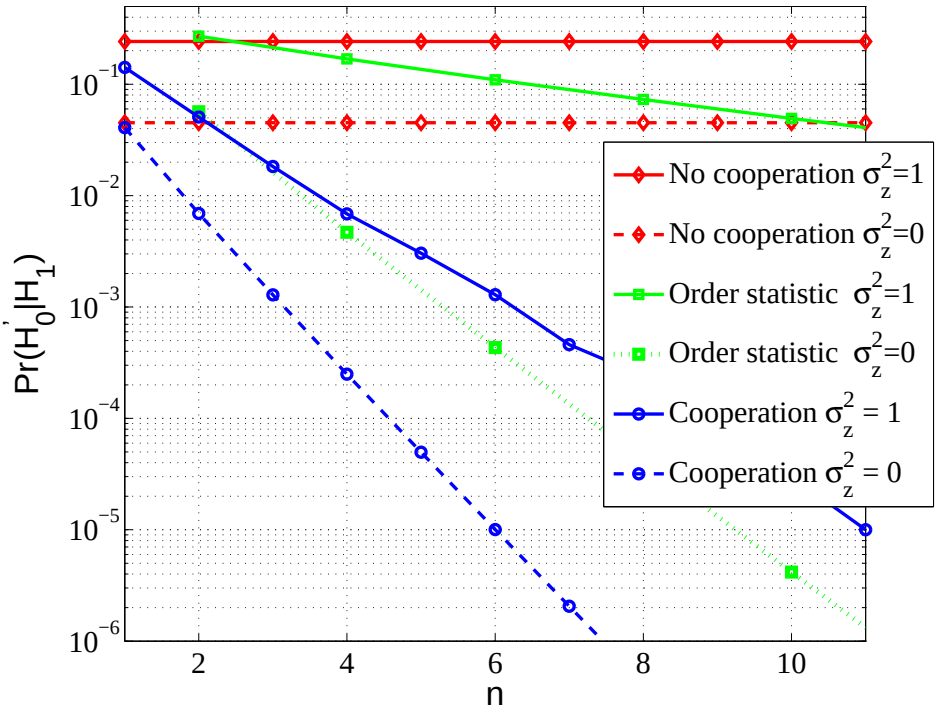


Figure 3.3: Probability of interfering with PU

$$m_{ij}^{(t)}(0) = \psi_{ij}(0,0)\phi_i(0)\prod_{k\in N(i)\backslash j} m_{ki}^{(t-1)}(0) + \psi_{ij}(1,0)\phi_i(1)\prod_{k\in N(i)\backslash j} m_{ki}^{(t-1)}(1) \tag{3.31}$$

$$m_{ij}^{(t)}(1) = \psi_{ij}(0,1)\phi_i(0)\prod_{k\in N(i)\backslash j} m_{ki}^{(t-1)}(0) + \psi_{ij}(1,1)\phi_i(1)\prod_{k\in N(i)\backslash j} m_{ki}^{(t-1)}(1) \tag{3.32}$$

The $\psi_{ij}(X_i, X_j)$ can be defined as

$$\psi_{ij}(X_i, X_j) = \begin{cases} 1, X_i = X_j \\ 0, \text{otherwise} \end{cases} \tag{3.33}$$

which captures the fact that $m_{ij}^{(t)}(0)$ should be influenced by the same type of belief, which are the $m_{ij}^{(t-1)}(0)$ from the other nodes. Therefore, we can simplify the equation as follows:

$$m_{ij}^{(t)}(0) = \psi_{ij}(0)\phi_i(0)\prod_{k\in N(i)\backslash j} m_{ki}^{(t-1)}(0) \tag{3.34}$$

$$m_{ij}^{(t)}(1) = \psi_{ij}(1)\phi_i(1)\prod_{k\in N(i)\setminus j} m_{ki}^{(t-1)}(1) \tag{3.35}$$

Let $m_{ij}^{(t)} = log\left(m_{ij}^{(t)}(0)/m_{ij}^{(t)}(1)\right)$, then we can drop the parameter x and consider its log-likelihood ratio as below

$$m_{ij}^{(t)} = \phi_i + \sum_{k\in N(i)\setminus j} m_{ki}^{(t-1)} \tag{3.36}$$

3.6.4 Convergence Analysis

Let M be a column vector of m_{ij} for all $i \neq j$; M is a matrix of size $1 \times l$, $l = 2|E|$. Then we can write (A.1) in a matrix form

$$M^{(t)} = \Phi + WM^{(t-1)} \tag{3.37}$$

where W is a $l \times l$ directed-adjacent matrix whose elements $w_{kl} = 1$ when m_l^{t-1} contributes to the k^{th} message m_k^t.

Φ is the initial value assignment from nodes to the edges.

$$\Phi = |A^-|\Phi_0$$

$|A^-|$ accounts for outgoing connection in edge-node incident matrix where the element is -1 when it's an out going edge and 1. In contrary, A^+ represents the

incoming part of the edge-node incident. Φ_0 is an $nx1$ vector represent the initial value on each nodes.

First step, M^0=Φ then

$$\begin{aligned} M^{(1)} &= \Phi + W\Phi \\ M^{(2)} &= \Phi + WM^{(1)} \\ &= \Phi + W(\Phi + W\Phi) \\ &= \Phi + W\Phi + W^2\Phi \\ M^{(t)} &= (I + W + W^2 + W^3 + ... + W^t)\Phi \end{aligned} \tag{3.38}$$

The W^t has an interpretation; an entry w_{ij} gives the number of walk of length t from i to j. Thus W^t, in this case, indicates how the local information Φ propagates from one m_l to another by using exactly t hops. It can be observed that in order for a local information to propagate through out the network, $t = d$ iteration is required, where d is the diameter of the connected graph.

In the context of distributed average consensus (DC), this can be viewed as the t-order distributed consensus algorithm, i.e. it is equivalent to storing t-1 previous iterations. It is shown that the high order has faster convergence rate [24]. In this instance, our BP algorithm is not exactly like the DC algorithm because the matrix W are formed differently and some parameters are different. Moreover, the ultimate goal of the algorithm is that we want each local soft information to propagate and weight equally throughout the network. Hence w_{ij} of any W^t should have the same weight; but the loopy graph may cause some messages to be over stimulated. DC, on

the other hand, has been proven to be converged.

Step aside from the DC context, let look at $(A.4)$ alone, ones may recognize this as a geometric series of metrics. There are useful theorems regarded the series [25].

Geometric Series of Matrices 1. *The geometric series of matrices* $\sum_{t=0}^{\infty} M^t$ *converges if and only if.* $|\lambda_i| < 1$ *for each eigenvalue* $|\lambda_i|$ *of* M*. Then we have*

- $|\lambda_i| < 1$ *implies* $(I - M)^{-1}$ *exists.*
- $\sum_{t=0}^{\infty} M^t = (I - M)^{-1}$.
- $\sum_{t=0}^{n} M^t = (I - M)^{-1}(I - M^n)$.

In general, the theorem holds when $|\lambda_i| = 0$. A graph G is an acyclic directed graph if and only if its adjacency matrix has all eigenvalues equal to zeros [26].

With this observation in mind, we show that if the network connectivity tree-like, then the BP algorithm converges. To show this first we have to transform node-node connectivity graph, which is undirected tree graph, into message - message connectivity. Use fig. 3.4 We provide sample and steps as the following:

- Replace edges in the original undirected with 2 directed edge labelled as m_{ij}. These two edges correspond to the messages passed from one node to another. Figure 3.6 depicts the transformed result.
- Construct a new graph where nodes are labelled as m_{ij} which is the one-on-one mapping from the message.
- Connect a direct link from m_{ki} to m_{ij}, $k \neq j$, aka, connect any incoming messages from surrounding node except the one that the message is pointing

to in the original graph. Then we got the final acyclic directed graph as in fig. 3.6.

The new graph is a acyclic directed graph which represents exactly the W matrix mention previously. There are some observations worth noting here.

- For any m_{ij} where i is a leaf node in the original tree graph, m_{ij} is source node, whereas the edges are those with outgoing direction only.
- For any m_{ji} where i is a leaf node in the original tree graph, m_{ji} is sink node, where all edges are incoming.
- The other messages m_{ij} $i, j \neq$ leaf node are considered intermediate nodes.

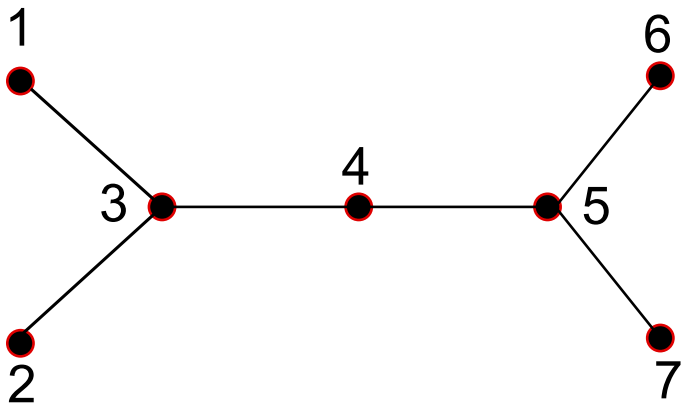


Figure 3.4: Example of a network with tree connectivity

Suppose we order the in the following order

$$[m_{13}, m_{23}, m_{65}, m_{75}, m_{34}, m_{54}, m_{43}, m_{45}, m_{32}, m_{31}, m_{57}, m_{56}]$$

The adjacency matrix W of the above example looks like this

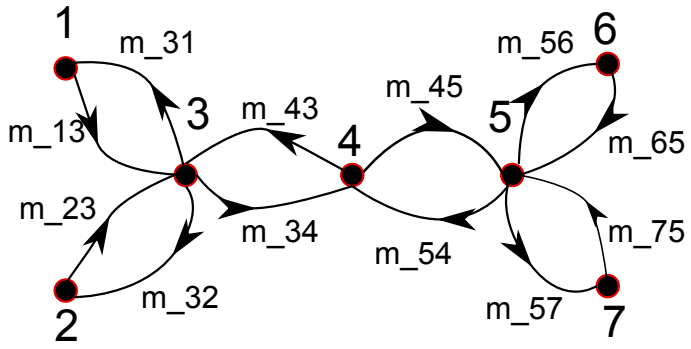


Figure 3.5: Second step in the transformation.

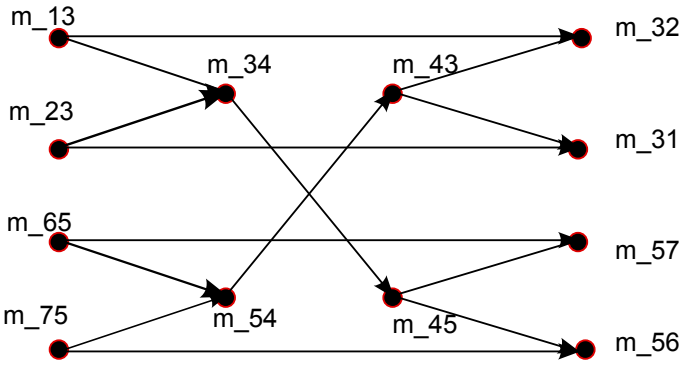


Figure 3.6: Second step in the transformation.

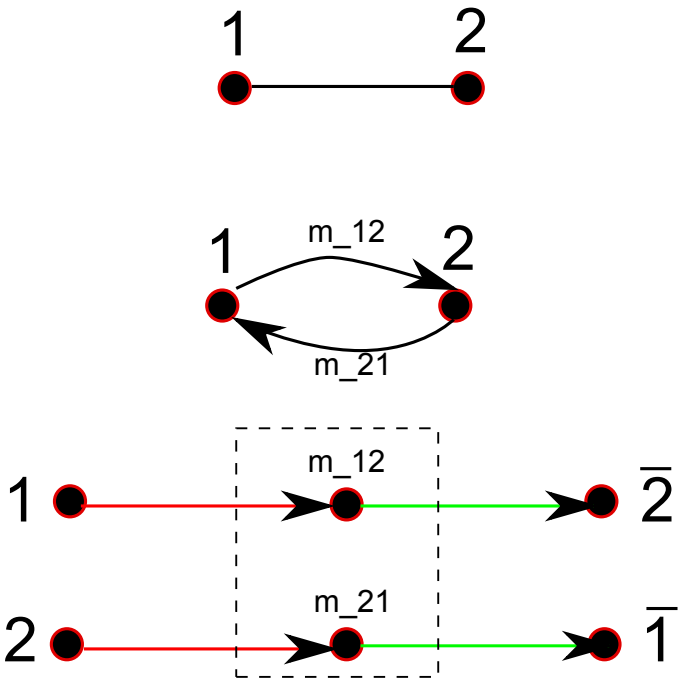


Figure 3.7: An example with network of size 2.

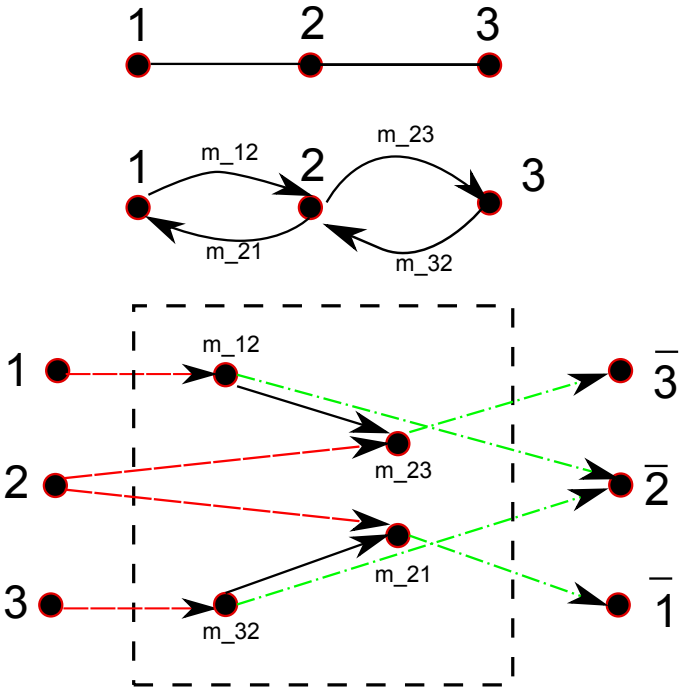


Figure 3.8: Another example with network of size 3.

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}$$

it's in the form of

$$\begin{bmatrix}
0 & 0 \\
C & 0
\end{bmatrix}$$

where C is $2l - f \times 2l - f$ where f is the number of leaf nodes and l is the number of edges. Since we are looking at tree graph, we know that $|E| = |n + 1|$ where n is the number of nodes. So the size of C can be written in term of nodes as $2(n + 1) - f \times 2(n + 1) - f$.

Claim: If a graph is tree-like, W is an adjacency matrix of acyclic directed graph and hence the BP algorithm converges.

Sketch Proof

1. *Consider an undirected tree graph, it can be converted to directed tree graph by replacing edge (i, j) by 2 edges m_{ij} and m_{ji}.*

2. *Construct another directed edge-to-edge graph $\hat{G}$ from eqn. (A.1), aka node m_{ij} and node m_{jk} in the new graph have a directed edge.*

3. *This new graph $\hat{G}$ is acyclic directed graph. The prove for this step is by induction, starting from a tree graph G of 2 nodes and then converted that graph to $\hat{G}$ and show that $\hat{G}$ has no cycle. Constructively add another pair of node and edge to G, G is still a tree and show that $\hat{G}$ still has no cycle.*

4. *Acyclic implies the eigenvalues, then from the theorem mention above, we*

5. *Hence the adjacency matrix W of $\hat{G}$ has $|\lambda_i| = 0$ and this proves the claim.*

Alternatively, one can show that $W^t = 0$ when $t > d$ where d is the diameter of the tree graph, hence $\sum_{t=0}^{\infty} W^t = \sum_{t=0}^{d} W^t = (I - M)^{-1}(I - M^d)$ converges.

Finally, the messages will be combined to make the final decision. It also takes the following form

$$B = \Phi_0 + \left(A^+\right)^T M$$

Next, we have to show that at the iteration $t \geq d$, each node's value will be equal to the sum of the initial values of all nodes. That is any element b_i in B will be equal to the sum of all element in Φ_o, e.g. the sum or all log-likelihood ratio. And this value is, in fact, equal to the optimal centralized algorithm, aka, the Equal Gain Combining (ECG).

Claim 2: any element b_i in B will be equal to the sum of all element in Φ_o, e.g.

$$\begin{aligned} B &= \Phi_0 + \left(A^+\right)^T M \\ &= \left(\left(\Phi_0\right)^T [111...1]_n\right) \left([111...1]_n\right)^T \end{aligned} \tag{3.39}$$

Sketch Proof

1. *any tree graph, there is only 1 path from a pair of nodes.*
2. *From the property of the node-to-node tree in figure 3.5, it implies that the value b_i will be connected to the value Φ_{0i} for any i at most 1 time. And it also implies that b_i is connected to all Φ_{0i}.*
3. *From $W + W^2 + ... + W^t$ mean that the propagated value Φ_0 from any node will either reach a node b_i in 1 iteration, 2 iterations .. t iterations respectively. It implies that Φ_0 must reach b_i at least 1 time.*
4. *Each Φ_0 reaching to b_i at least 1 and at most 1 time means it reaches b_i exactly 1 time.*
5. *Hence, the final value of each element in B is in fact the sum of the initial value from each node. And this concludes the claim.*

3.6.5 Numerical and Computational Example

We use fig. 3.7,3.8 and 3.5 as examples. This will give the reader a clearer picture on how the algorithm works distributively as a whole. We show both matrix and

graphical forms. First let set the initial values Φ_n to be all 1. Then

$$\Phi_l = |A^-|\Phi_n$$

is

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Since $W = 0$ we do not need any iteration, we can evaluate B right away, which is.

$$B = \Phi_n + \left(|A^+|\right)^T (I + W + W^2 + W^3 + ... + W^t)|A^-|\Phi_n \tag{3.40}$$

$$= \Phi_n + \left(|A^+|\right)^T \Phi_n \tag{3.41}$$

$$= (I + \left(|A^+|\right)^T)\Phi_n \tag{3.42}$$

$$B = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \right) \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{3.43}$$

$$= \begin{bmatrix} 2 \\ 2 \end{bmatrix} \tag{3.44}$$

Next we see the matrix for the tree of size 3.

$$B = \Phi_n + \left(|A^+|\right)^T (I + W + W^2 + W^3 + ... + W^t)|A^-|\Phi_n \tag{3.45}$$

$$= \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \right) \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{3.46}$$

$$= \begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix} \tag{3.47}$$

We write down the computation required in each node in the table 3.1, 3.2 and 3.3 respectively.

Table 3.1: Intermediate Numerical State At Each Iteration

Message no.	Equation	0	1	2
m_{12}	ϕ_1	1	1	1
m_{21}	ϕ_2	1	1	1
Belief	Equation			
b_1	$\phi_1 + m_{21}$	0	2	2
b_2	$\phi_2 + m_{12}$	0	2	2

3.6.6 Improved Belief Propagation based on weight

In case of loopy undirected graph, W may not always be the case. However, if we modify our matrix by adding some weight to it; let $W_d = W/d_m$, where d_m is the maximum degree of the graph. We know that [21], for adjacent matrix, $|\lambda_i| \leq d_m$.

Table 3.2: Intermediate Numerical State At Each Iteration

Message no.	Equation	0	1	2
m_{12}	ϕ_1	1	1	1
m_{23}	$\phi_2 + m_{12}$	1	2	2
m_{32}	ϕ_3	1	1	1
m_{21}	$\phi_2 + m_{32}$	1	2	2
Belief	Equation			
b_1	$\phi_1 + m_{21}$	0	3	3
b_2	$\phi_2 + m_{12} + m_{32}$	0	3	3
b_3	$\phi_3 + m_{23}$	0	3	3

Table 3.3: Intermediate Numerical State At Each Iteration

Message no.	Equation	0	1	2	3	4
m_{31}	$\phi_3 + m_{23} + m_{43}$	1	3	4	6	6
m_{13}	ϕ_1	1	1	1	1	1
m_{23}	ϕ_2	1	1	1	1	1
m_{32}	$\phi_3 + m_{13} + m_{43}$	1	3	4	6	6
m_{43}	$\phi_4 + m_{54}$	1	2	4	4	4
m_{34}	$\phi_3 + m_{13} + m_{23}$	1	3	3	3	3
m_{45}	$\phi_4 + m_{34}$	1	2	4	4	4
m_{54}	$\phi_5 + m_{65} + m_{75}$	1	3	3	3	3
m_{56}	$\phi_5 + m_{45} + m_{75}$	1	3	4	6	6
m_{65}	ϕ_6	1	1	1	1	1
m_{75}	ϕ_7	1	1	1	1	1
m_{57}	$\phi_5 + m_{45} + m_{65}$	1	3	4	6	6
Belief	Equation					
b_1	$\phi_1 + m_{31}$	0	4	5	7	7
b_2	$\phi_2 + m_{32}$	0	4	5	7	7
b_3	$\phi_3 + m_{13} + m_{23} + m_{43}$	0	5	7	7	7
b_4	$\phi_4 + m_{34} + m_{54}$	0	7	7	7	7
b_5	$\phi_5 + m_{45} + m_{65} + m_{75}$	0	5	7	7	7
b_6	$\phi_6 + m_{56}$	0	4	5	7	7
b_7	$\phi_7 + m_{57}$	0	4	5	7	7

Then the eigenvalues of $W_d = |\lambda_i|/d_m \leq 1$. Hence, the series

$$M^{(t)} = \Phi + \frac{W}{d} M^{(t-1)} \tag{3.48}$$

$$M^{(t)} = (\frac{I}{d^0} + \frac{W}{d^1} + \frac{W^2}{d^2} + ... + \frac{W^t}{d^t})\Phi \tag{3.49}$$

converges.

So adding weight to the message is one possibility. Despite of its rigorous close form, the performance is not promising. Consider the term $\frac{W}{d}^t$, which corresponds to the messages from t-hop neighbors, it is weighted so small.

Another possibility is to adding weight in an opposite way, for example, consider this equation below

$$M^{(t)} = \frac{\Phi}{d^t} + W M^{(t-1)} \tag{3.50}$$

$$
\begin{aligned}
M^{(1)} &= \frac{\Phi}{d^1} + W\Phi \\
M^{(2)} &= \frac{\Phi}{d^2} + WM^{(1)} \\
&= \frac{\Phi}{d^2} + W(\frac{\Phi}{d^1} + W\Phi) \\
&= \frac{\Phi}{d^2} + W\frac{\Phi}{d^1} + W^2\Phi \\
M^{(t)} &= (\frac{I}{d^t} + \frac{W}{d^{t-1}} + \frac{W^2}{d^{t-2}} + ... + \frac{W^t}{d})\Phi
\end{aligned} \tag{3.51}
$$

The combination of (3.48) and (3.50) is also possible. Or a more general form can be expressed as

$$
M^{(t)} = h(.)\frac{\Phi}{f(.)} + i(.)\frac{W}{g(.)}M^{(t-1)} \tag{3.52}
$$

where $f(.)$ and $g(.)$ refer to arbitrary weight functions while $h(.)$ and $i(.)$ can be any sign functions in hope that they would help cancelling excessive stimulated messages caused by loopy structure of the sensing network.

Nevertheless, we simulate the BP, at this time, we focus on how well the value of BP converge to the value of EGC or MGC (EGC as well, is viewed as the target of DC algorithm).

We compare the BP and DC algorithm to see their convergence rates. BP has faster convergence rate but also experiences the error floor while DC slowly approaches the centralized MRC, which is laid as a lower bound for the distributed algorithm. Fig. (3.9) and (3.10) depict the evaluation of DC and BP on a square network of size

$m \times m$.

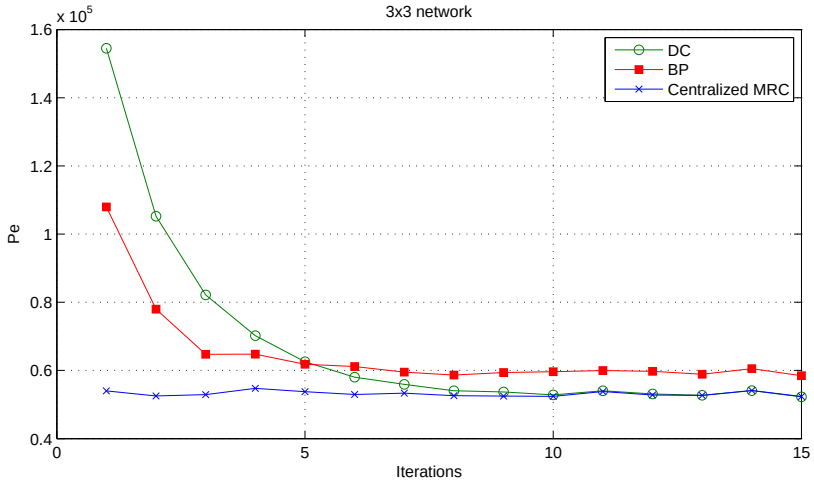


Figure 3.9: Convergence rate of BP, DC and MRC at 3x3 network.

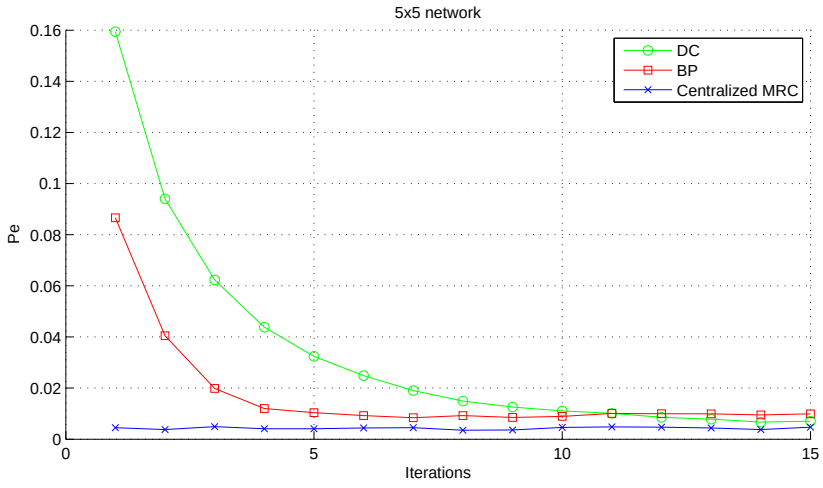


Figure 3.10: Convergence rate of BP, DC and MRC at 5x5 network.

From above plots, one may come up with a hybrid approach, e.g. performing BP on the first half of the iterations following by the DC algorithm. The results are shown and compared below in fig. (3.11).

Figure 3.12 shows the comparison between a loopy grid network and a tree-like

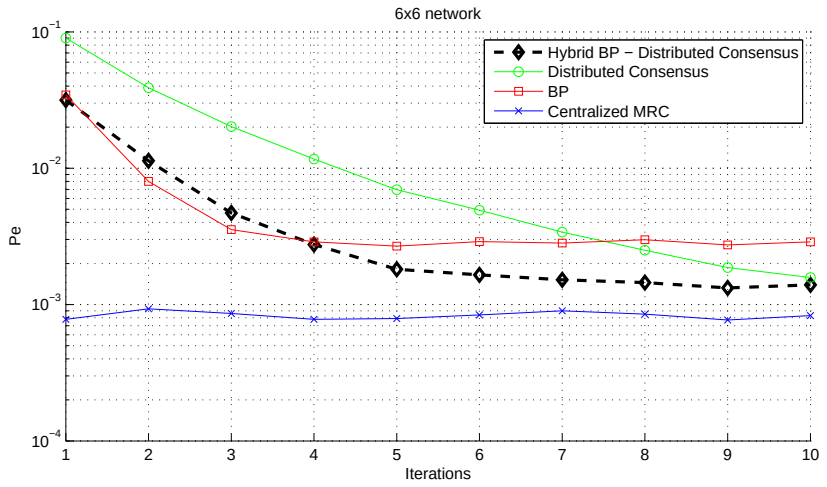


Figure 3.11: Convergence rate of BP, DC and MRC at 6x6 network.

network of size 7x7. The tree is formed by trimming all excessive edges, resulted in tree with diameter of 14. The BP algorithms running on this tree approaches the lower bound and out perform the BPs on loopy networks. It converges a little slower because the information cannot flow through shot-cut edges that have been removed. DC, on the other hand, does not work well with tree-like structure.

In figure 3.13, the simulation results of eqn. (3.49) and (3.51) are compared with other distributed schemes on a network of size 10×10.

We simulate both BP and DC algorithm on i.i.d. Rayleigh fading channel with noise $\sigma = 2$ of a network of size 25 nodes. We formulate the network into 4 structures, 5x5 square grid, random placement in unit square, 5x5 grid transformed to tree graph and finally random placement transformed to tree. We use DMST to transform the original graph into tree. An instance of a random graph we use is depicted in fig. 3.1.

Figure 3.14 shows that the optimal choice of the parameter α for DC algorithm always gives the best rate of convergence.

Figure 3.15,3.16 and 3.17 show the result from standard BP algorithm and optimal

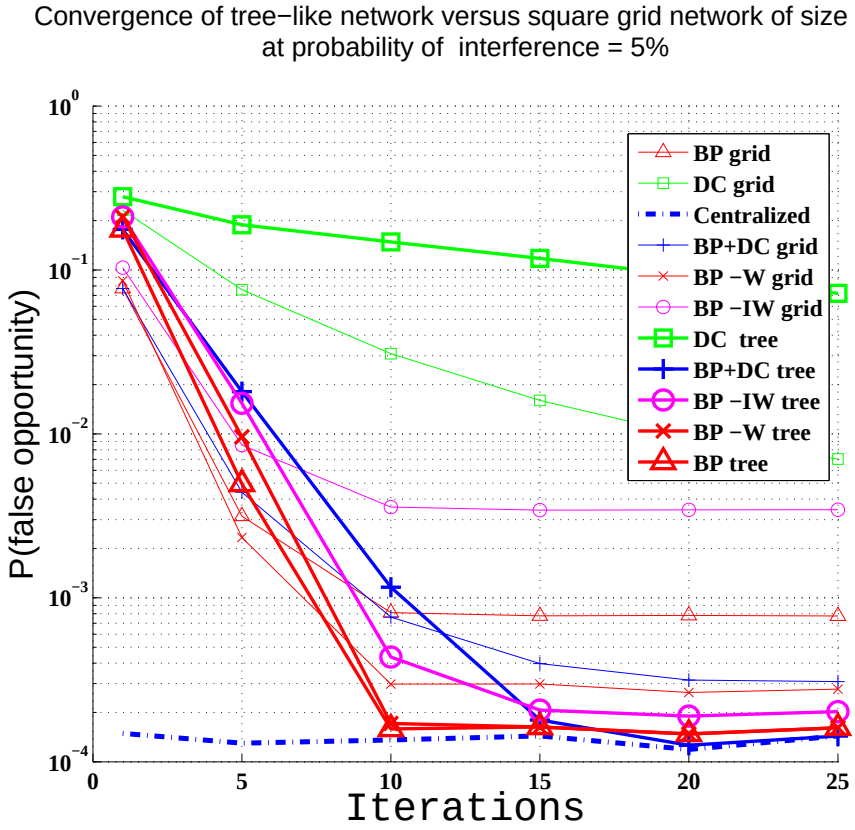


Figure 3.12: Comparing distributed algorithms in tree-like network and loopy network.

DC algorithm as well as weighted BP algorithms (BP-W for equation (3.48) and BP-IW for equation (3.50)). In grid network, BP provides faster convergence rate but hit the error floor, while DC eventually converges to the optimal result. When transforming the grid into tree graph, however, BP gives the optimal result exactly in 10 iterations (diameter of 5x5 network is 10). Comparing both original and the modified tree, BP in tree still gives the best performance followed by DC when operate in original network. BP-W and BP-IW are poor comparing to the standard BP and DC algorithms.

Similarly, 3.18,3.19 and 3.20 show the result when operate in a random network.

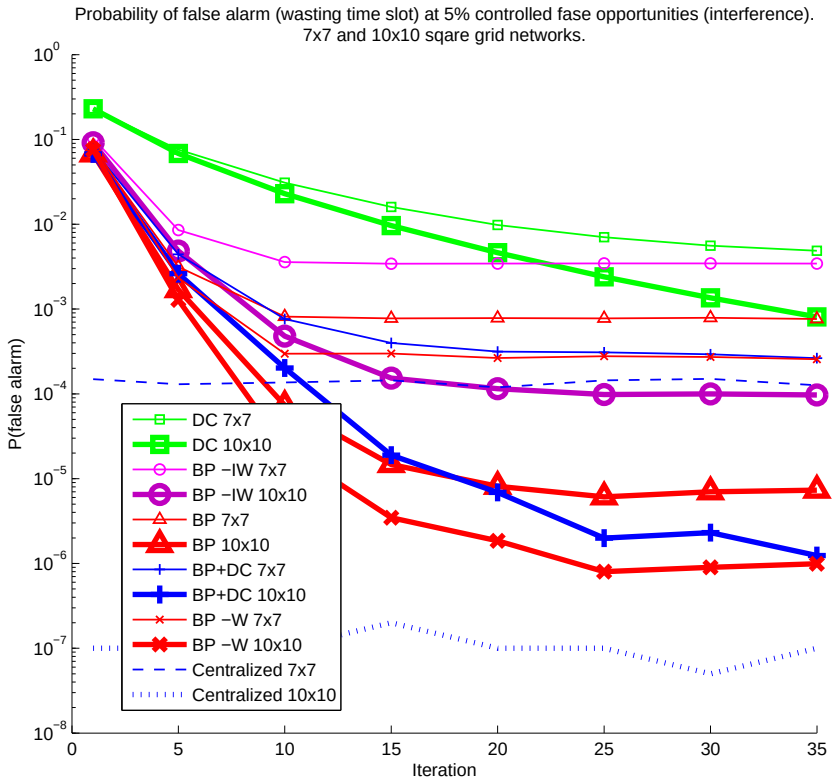


Figure 3.13: Convergence rate of BP, DC, BP with weights and hybrid BP+DC on 7x7 and 10x10 square networks.

With it original network connectivity BPs family performs poorly due to the number of short loop while DC benefits from the higher node degree and short diameter. However, BP will perform best when transform this graph into tree where it reaches the optimal point within 4 iterations. Thanks for the shorter diameter and the tree structure that has no loop in it.

The simulation results confirm that DC performs well in dense networks while BP is better in sparse networks, especially when it is tree-like network.

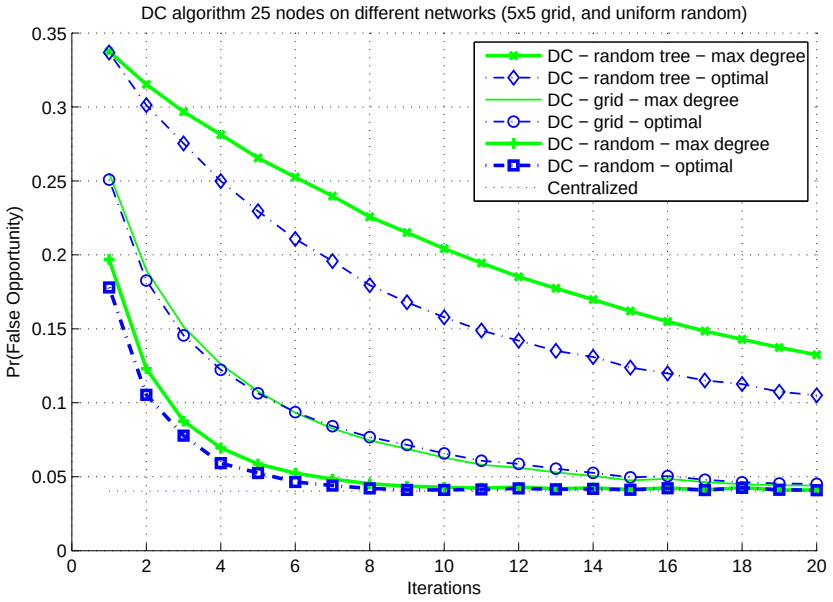


Figure 3.14: Convergence rate of DC algorithm with different α parameters.

3.7 Simulation Results

Extensive simulations are performed to evaluate the proposed cooperative sensing strategies. Take the same 10-secondary-user topology in Figure 3.1 as an example. Assuming that the PU is transmitting at a normalized power of 1dB and is 10 kilometer away from the SUs. The signal propagates and reaches the SUs obeying the path loss model in (4.1) and (4.2), with $d_0 = 100$ m, path loss exponent $\alpha = 3$. The varience of the long-normal shadowing may vary from $\sigma_w^2 = 1$ dB up to 20 dB.

We compare several cooperative strategies, including the distributed strategies using BP, majority-voting (hard-combining), equal-gain combining and maximal-ratio combining, as well as the centralized strategies using majority-voting, EGC and MRC. For comparison, we also include the case of no cooperation. Three iterations are performed with the BP algorithm.

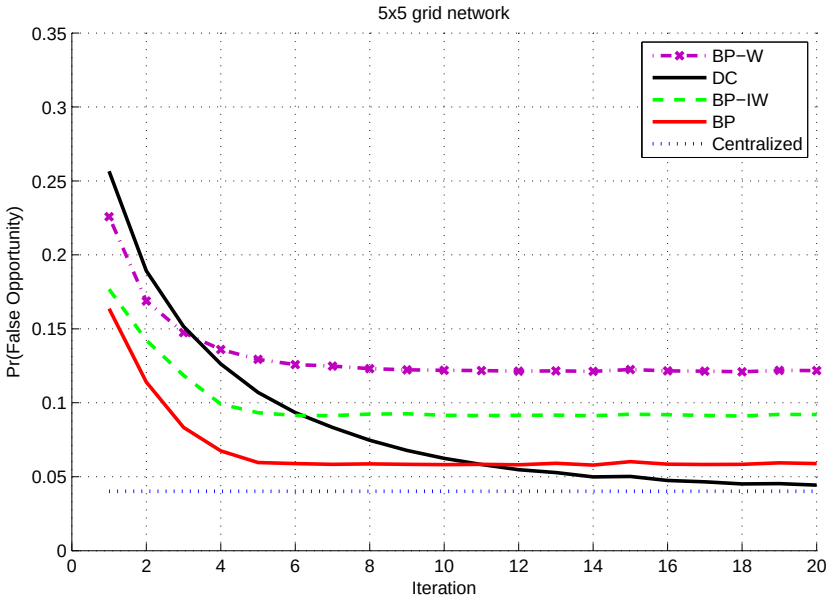


Figure 3.15: Convergence rate of 5x5 grid network

We first evaluate the distribution of the belief without applying the decision threshold θ. Figure 3.21 plots the pdf of the belief when the PU is active and when the PU is idle. We assume that sufficient time-averaging has been performed, such that the impact of noise uncertainty is negligible, but the channel is dominated by heavy shadowing.

The result clearly indicates the remarkable improvement in the amount of belief enabled by BP, comparing to the non-cooperative case, whose belief concentrates heavily at 0.5 and will therefore cause a heavy tailing probability.

Figure 3.22 compares the sensing performance for different strategies. The x-axis denotes the different shadowing effect from weak to strong $\sigma_w^1 = 1$ to 9 dB, and the y-axis denotes the probability of missed opportunity. The curves come in four groups. The top-most (worst-performing) group is the case of no cooperation.

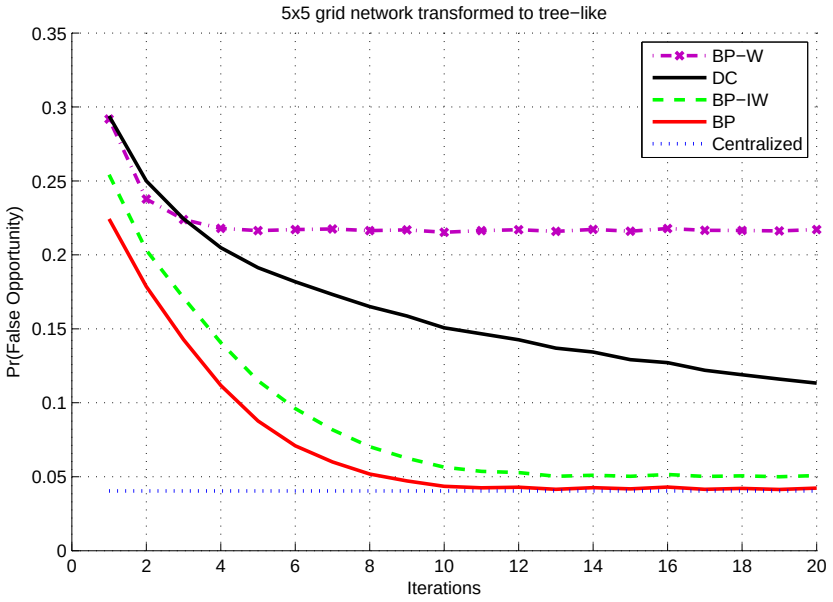


Figure 3.16: Convergence rate of 5x5 grid network when transformed to tree graph.

The next group includes the degenerated case of the distributed cooperation, such majority vote, EGC and MRC. The third group is BP, which outperforms all the other distributed algorithms. Finally the centralized EGC and MRC deliver the best performance.

The performance of a receiver can also be illustrated using its receiver operating characteristic (ROC) [19] which is generally demonstrated as the plot of the detection probability against the false alarm rate. Since the normalized belief is within the range of 0 and 1, we set the decision threshold θ from 0.1 to 0.9, and evaluate the actual detection and false alarm probabilities of different algorithms as well as their sensitivity to the threshold. The results are shown in Figure 3.23.

Again, we see that the performances come in four groups, where BP outperforms

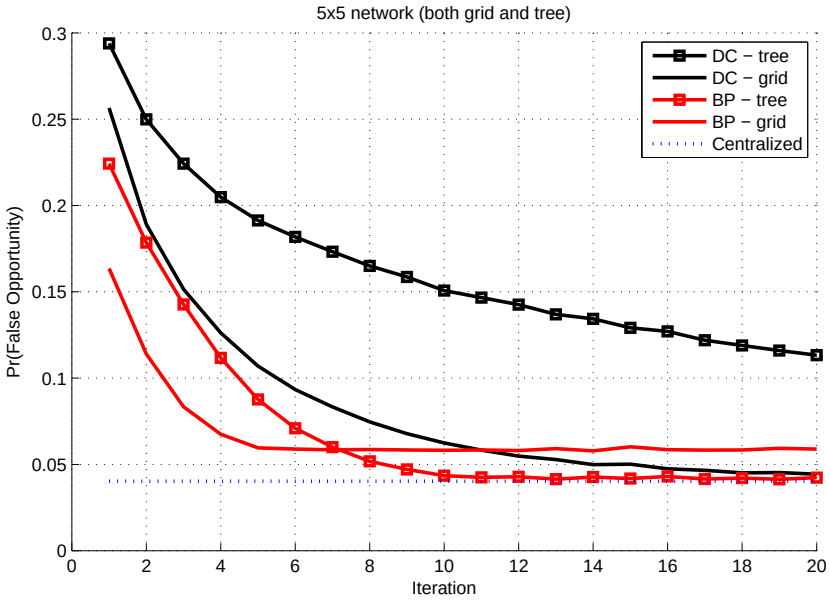


Figure 3.17: Convergence rate of 5x5 grid network. Both original network and tree network are compared.

the rest of the distributed algorithms, but falls slightly short of the centralized algorithm; nevertheless, no-cooperation lags far behind.

3.8 Chapter Conclusion

We have carefully evaluated spectrum sensing through probabilistic inference operated naturally on the network graph. Through analysis and simulations, we show effective cooperation can drastically improve the sensing accuracy. We conclude by promoting belief propagation as a feasible and effective distributed algorithm for cooperative spectrum sensing.

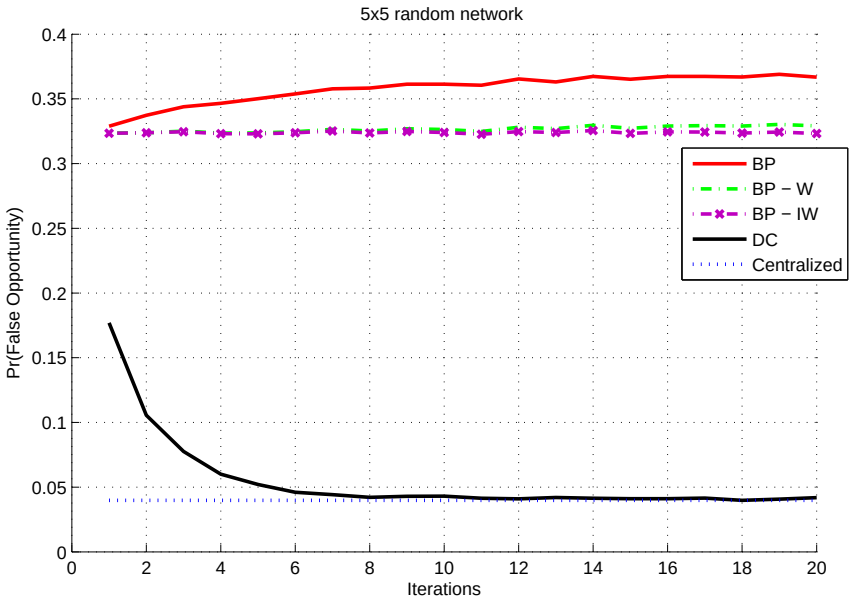


Figure 3.18: Convergence rate of 5x5 random network.

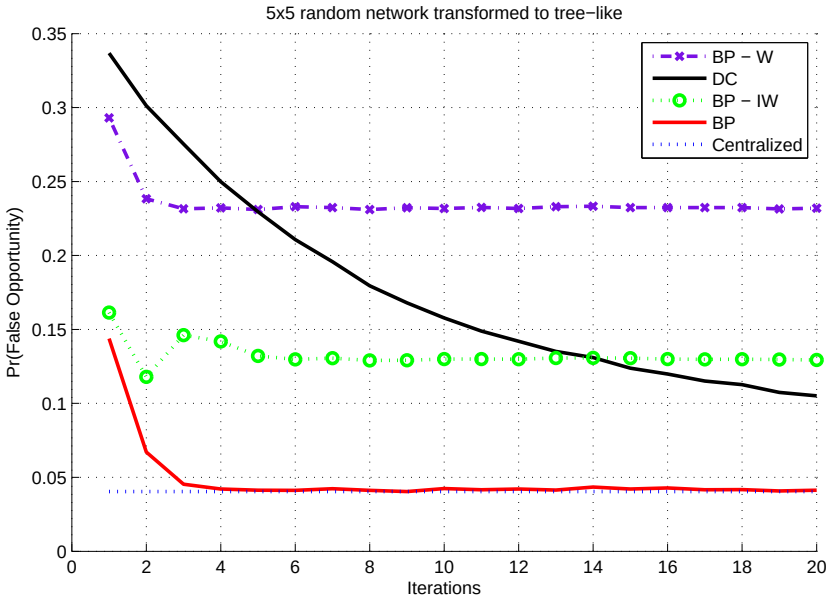


Figure 3.19: Convergence rate of 5x5 random network when transformed to tree network.

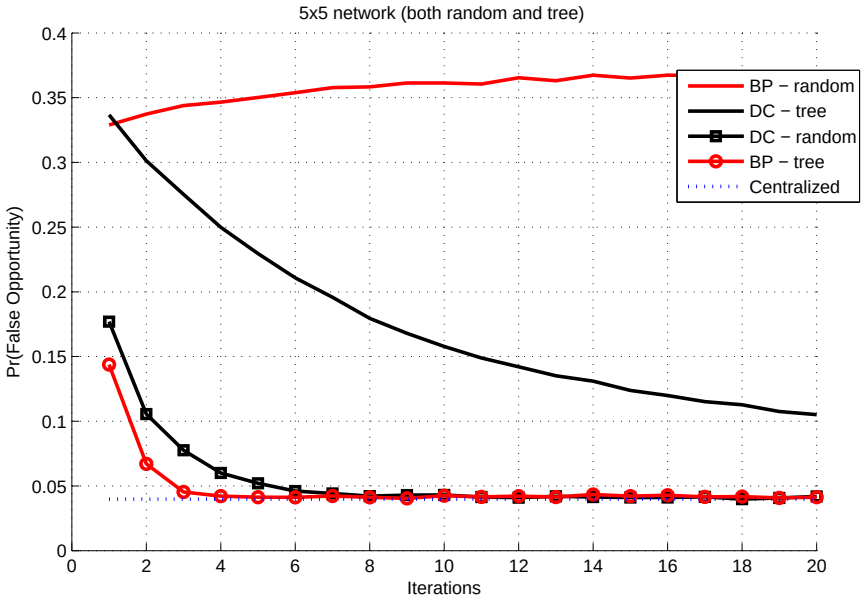


Figure 3.20: Convergence rate of 5x5 random network, both original network and after transformed to tree.

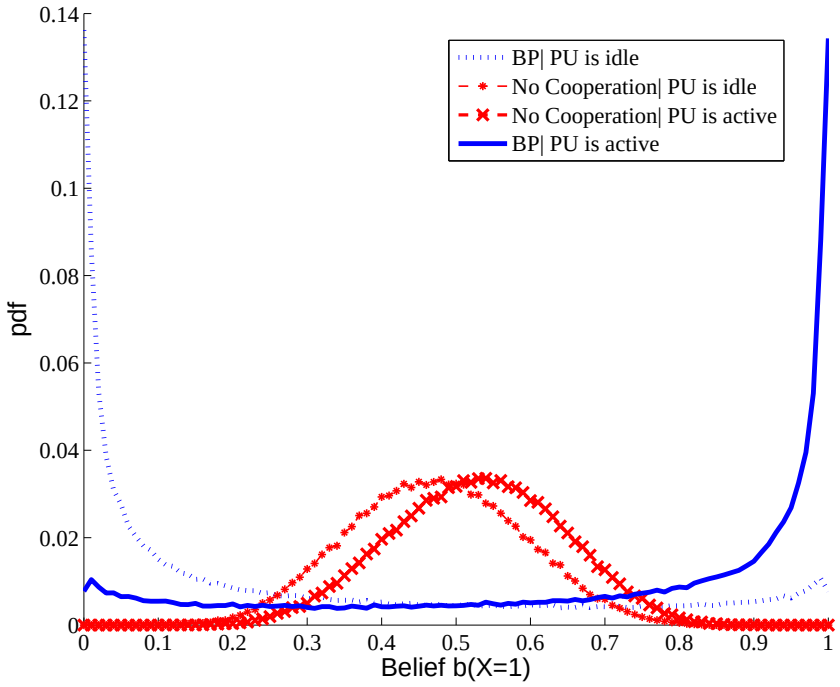


Figure 3.21: The average pdf of the amount of belief that PU is active $b(X = 1)$ given PU is active/idle at σ_w^2=20 dB.

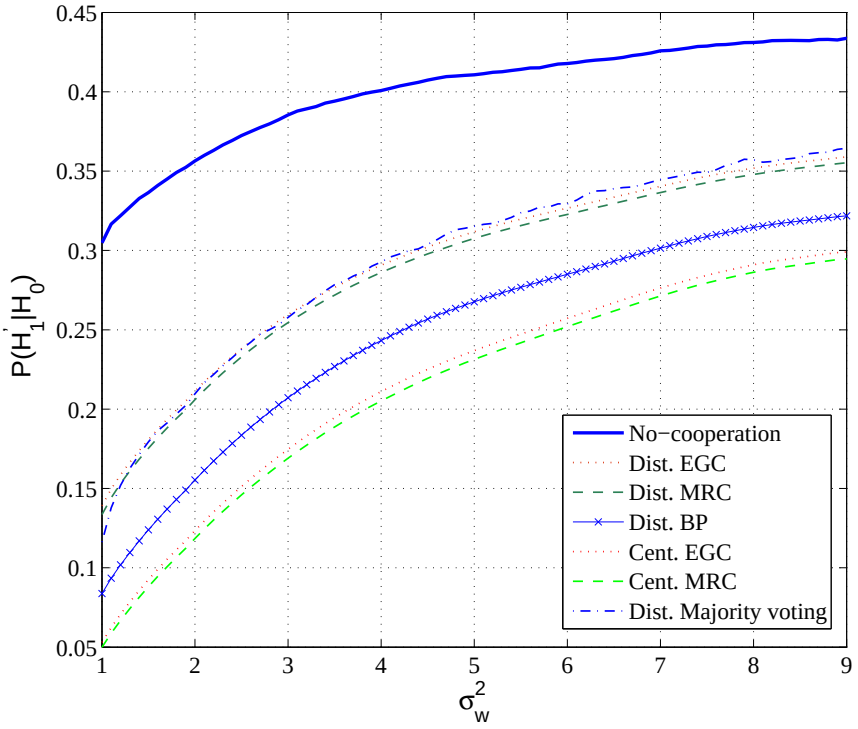


Figure 3.22: Performance comparison with different shadowing effect $\sigma_w = 1$ to 9.

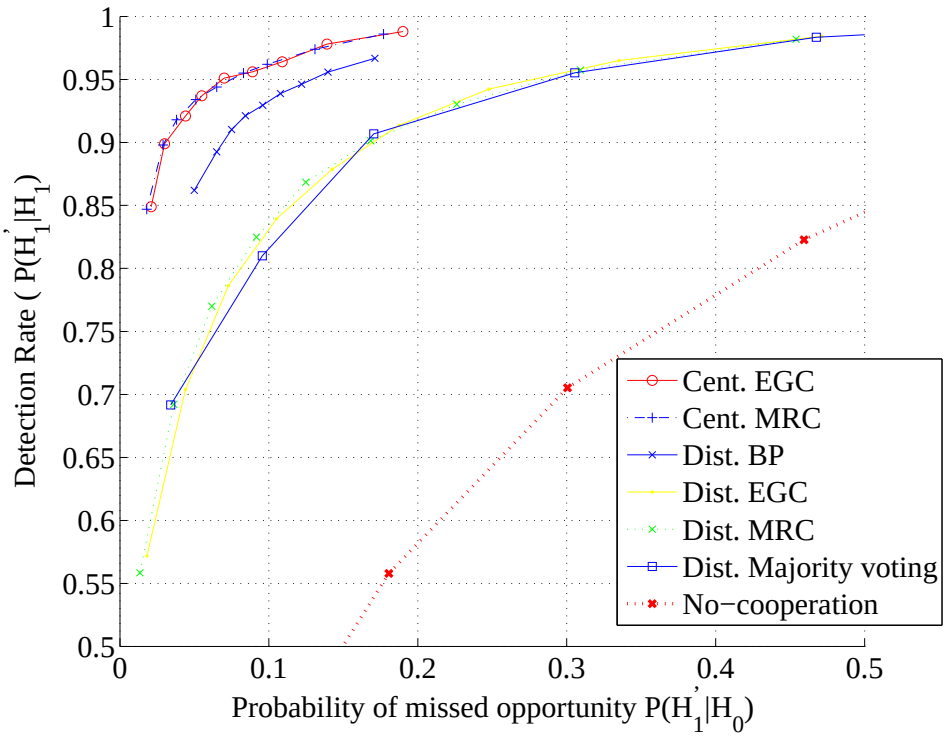


Figure 3.23: ROC comparison for different algorithms, $\sigma_w^2 = 1$.

Chapter 4

Cooperative Spectrum Sensing with Tri-State Probabilistic Inference

Cooperation can significantly improve the diversity order and hence the sensing accuracy in spectrum sensing in cognitive radio systems. Since cooperation inevitably introduces communication overhead, question arises as how and how much cooperation should be induced to attain the "low-hanging fruits" without being excessive and overshadowing the gain. Based on the topology graph, this paper proposes a distributed tri-state probabilistic inference mechanism for cooperative sensing. The conventional decision fusion strategies pull together all the local information (e.g. yes or no for some hypothesis), irrespectively of its quality, in the neighborhood.

The new idea in the tri-state decision fusion is that if a cognitive radio is rather unsure (up to a threshold) about its sensing result, then instead of sending out this

information (which may well be useless anywhere), it might as well remain silent, that is, stay in the third state of "erasure" to save energy (and bandwidth). Information-theoretic analysis is conducted to determine the optimal threshold that maximizes the date-rate to energy ratio. Extensive simulations are computed to verify the theoretical results.

4.1 Introduction

Cognitive Radio (CR) is one example of upcoming trends where allocated spectrum can be temporarily assigned to secondary users (SU) when the spectrum is idle, to allow a huge opportunity in enhancing the spectrum usage. The fundamental challenge here is how secondary users can use the spectrum without interfering with primary users (PU). Hence, the first tool to be included in CR architecture is to detect the spectrum usage, known as Spectrum Sensing.

It is a key component in the architecture for future opportunistic wireless communication. *Cooperative* spectrum sensing is an attractive method that provides promising detection performance and robustness against the nature of wireless propagation such as fading from multipath and shadowing effects [29]. In many scenarios where multiple SUs are located in the same region, it is natural for these units to jointly share the locally sensed information among themselves and perform decision fusion.

A large number of cooperation techniques have been proposed [29]-[36]. These

techniques can be classified in several ways. For example, from the computing perspective, there are centralized schemes where a fusion center is deployed for centralized control and computing [32, 33], and distributed schemes where SUs pull together information in the neighborhood cooperate and individually perform reasoning and decision making [34, 35].

In terms of data format, some use soft information in the form of probabilities or log-likelihood ratios (LLR) or the like, to give finer results at the cost of a higher computation cost and power consumption (when exchanging such information) [33]. Hard information schemes, or, quantized information, use fewer bits to represent data being exchanged, and hence consume a lower computation cost and less communication energy [29]. Cooperative spectrum sensing generally operates in a peer-to-peer fashion, but for large systems, cluster-based schemes are also proposed [30, 31].

In addition to ad-hoc decision fusion, recently, a solid method emerges from belief propagation algorithm is also being studied [32][36]. The method in [32] uses a fusion center which, after pulls together the preliminary detection results, formulates the decision scheme as an iterative message-passing algorithm on a factor graph that is irrelevant to the network topology. The scheme in [36] extends the centralized computation to a fully distributed belief propagation, whose supporting graph comes directly from the topology graph.

The network graph is further viewed as Markov Random Fields (MRF), on which iterative message changing and computation are performed in a decentralized manner. Specifically, at each iteration, an SU processes belief messages collected from its immediate neighbors and sends back the updated beliefs (possibly different beliefs to different neighbors). for propagates it own processed message to the others SUs. The

algorithm runs iteratively until a certain number of rounds is reached or the belief values converge (e.g. no significant change from the previous iteration).

Although belief propagation provides efficient, effective and practical approach, exchanging soft information consumes a rather high amount of energy per transmission. To save energy, hard information may be adopted and majority voting or counting-based method may be used in decision fusion at each local SU.

Since cooperation inevitably introduces communication overhead, we ask the question of how and how much cooperation should be induced to attain the "low-hanging fruits" without being excessive and overshadowing the gain. The conventional decision fusion strategies collect all the messages available in the neighborhood, irrespectively of their quality. This can be wasteful, since not all messages are equally informative and some may be practically useless (e.g. a probability of around 0.5 or an LLR of around 0 for two-hypothesis decision).

In the case of hard message exchange, the more energy-economic case than soft message exchange, the problem can be even worse, since a decision of 1 or 0 may be drawn with a high confidence level or a very low one. To make message exchange effective and especially to save energy, here we propose a distributed tri-state probabilistic inference mechanism. The idea is that if a cognitive radio is rather unsure (up to a threshold δ) about its sensing result, then instead of sending out this information (which may well be useless anywhere), it might as well remain silent, that is, stay in the third state of "erasure" to save energy (and bandwidth). In the other words, we explicitly consider the density of reliability per unit energy in the message exchange.

In addition to direct energy saving (due to skipped transmission), the system may

also benefit from a finer quantization resolution, since ternary input is better than binary input (the conventional hard message exchange). Information-theoretic analysis is conducted to determine the optimal threshold that maximizes the information rate per unit energy, and to quantify the amount of energy saving. Extensive simulations are computed to verify the theoretical results.

This paper is organized as follows. The system model and problem formulation is presented in section 4.2. The system analysis is given in section 4.4. Section 4.5 shows the simulation results. Finally, we give the conclusion in section 4.6

4.2 System Model and Problem Formulation

Consider a set of n secondary users scattered near the guard band of a primary user, forming a network topology. We consider a complex and practical wireless channel model that is dominated by large-scale shadowing (and random fading) along with log-distance path loss.

Let d_i be the distance between the PU and the ith SU. The instantaneous channel gain $h(d_i)$, can be modelled as the combined effect of an average path loss and a lognormal shadowing/fading effect, which, when measured in decibel (dB), becomes normal distributed:

$$h(d_i) = \bar{h}(d_0) + 10\alpha \log_{10}(d_i/d_0) + w, \quad (dB) \tag{4.1}$$

where $\bar{h}(\cdot)$ stands for the average path loss, d_0 is some close-in reference distance, α is the path loss exponent that generally takes value between 2 to 4, and w (in dB) is

a Gaussian random variable with zero mean and variance σ_w^2, subsuming the factors of shadowing and fading. The sensed signal y_i at the ith SU is given by

$$y_i = h_i x_i + z_i, \tag{4.2}$$

where $h_i = h(d_i)$ is lognormal, and z_i is the additive white Gaussian noise with zero-mean and variance σ_z^2.

Spectrum sensing involves determining one of following two hypotheses

$$H_0 : \text{The target spectrum is currently idle.} \tag{4.3}$$

$$H_1 : \text{The target spectrum is currently active.} \tag{4.4}$$

The decision may be made independently by each SU. For better accuracy, cooperation can be arranged to allow a group of co-located SUs leverage sensing results of each other. Effective decision fusion can be performed on *soft* information, namely, the likelihood ratio test $p(H_0|y)$ versus $p(H_1|y)$, where y is a random variable or a random vector denoting the sensed signal power by the SU.

The decision of whether the target spectrum is idle for re-use at the ith SU can be made via Δ-accuracy detection which guarantees that the probability of *false opportunity* (i.e. interfering with the PU), $p(H_0|H_1)$, does not exceed Δ. Let θ be the critical threshold for interfering with the PU. We decide this piece of spectrum is re-usable if $\frac{p(H_1|y)}{p(H_0|y)} < \theta$, and unavailable otherwise. At any desire value of δ, the threshold θ can be calculated via the following relation:

$$\Delta = \int_{y:\frac{p(H_1|y)}{p(H_0|y)}<\theta} p(y|H_1)dy, \tag{4.5}$$

This method is known as the Neyman-Pearson method [20]. Another important assessment for sensing and detection is the probability of *missed opportunity*, $p(H_1|H_0)$. It is desirable to have both false opportunity and missed opportunity under control, but since they are two competing factors, compromise is usually made to best satisfy the application requirement.

4.2.1 SU-to-SU Quantized Information

Assuming that the secondary user knows the channel state information and uses energy detection mechanism to make initial hard decision. The following normalized channel model $y_p = x_p + n_p$ is used, where $x_p \in \{0, 1\}$, n_p is additive white Gaussian noise (AWGN) with variance σ_p^2. Using the rational that when the sensed result is too ambiguous for any confident conclusion, then that secondary user should refrain from sharing this info, since the (energy) cost of transmitting this piece of information overweights the value of the information itself.

Thus, the sensed information will be quantized into 3 symbols using this quantization function,

$$g(y) = \begin{cases} -1 & y \leq -\delta \\ 0 & -\delta < y < \delta \\ 1 & y \geq \delta \end{cases}. \tag{4.6}$$

$$g(y) = \begin{cases} -1 & y \leq 0 \\ 1 & y \geq 0 \end{cases}. \tag{4.7}$$

where the middel level "0" is actually an erasure that will *not* be physically transmitted, and the other two levels are transmitted the same way as in the conventional hard information sharing and fusion schemes.

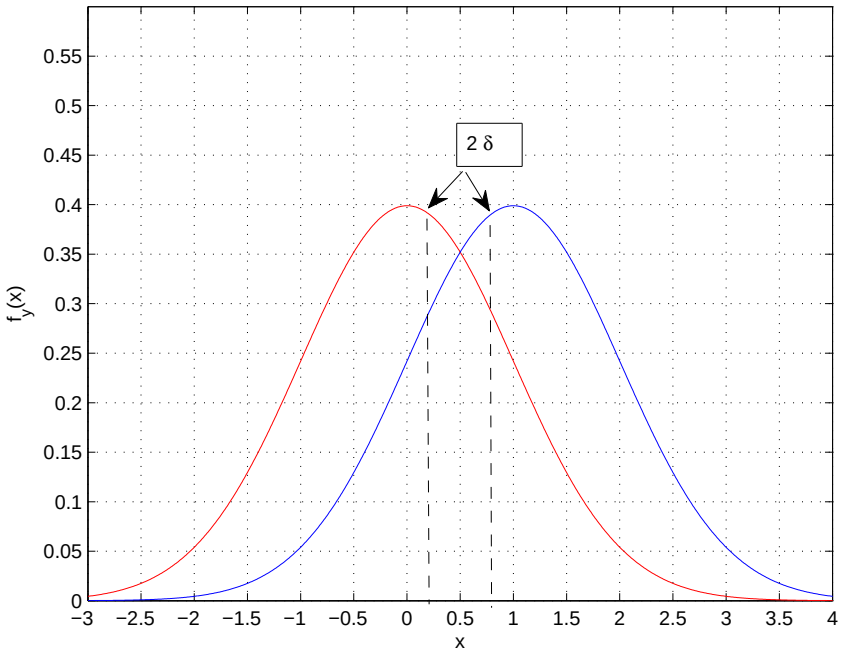


Figure 4.1: Primary User to Secondary User Channel Model

Then the above quantized values are transmitted among secondary users where the SU-to-SU channels are assumed to be AWGN channel with noise variance σ_s^2.

The next section, we describe how the quantized messages are used in the context of belief propagation. We start with the general belief propagation and then continue with our customized version.

4.3 Energy Efficient Belief Propagation with Hard Information

In chapter 3, the version of Belief Propagation uses LLR as the soft information. Consider the message passing part of the algorithm in the equation A.1 from the previous chapter and stated again below

$$m_{ij}^{(t)} = \phi_i + \sum_{k \in N(i) \backslash j} m_{ki}^{(t-1)} \tag{4.8}$$

In the conventional belief propagation, ϕ_i, m_{ki} are soft information. In this version, we substitute ϕ_i with the quantized value from eqn. (4.7). The iteration is performed normally and the decision is made in the same fashion. We will show the systematic way to determine the threshold in eqn. (4.7) in the next section.

4.4 System Analysis

In this section, we evaluate the capacity of our quantization method. We show that the mutual information is a function of σ_p and σ_s, in the other words, the SNR of primary user channel and the SNR of secondary user channel. Then the capacity is achieved by picking δ that maximize the mutual information. We start with the capacity in the next sub section followed by the method of finding δ. This δ will be used in the belief propagation algorithm described previously and the result will be shown later on.

4.4.1 Capacity of Ternary input discrete-time continuous output AWGN channel

Let P_0, P_1 and $P_\epsilon = 1 - (P_0 + P_1)$ be the probability that $g(y) = -1, 1$ and 0 respectively, we have

$$\begin{aligned} P_0 = P_1 &= \frac{1}{2}\int_{0.5+\delta}^{\infty} N(1, \sigma_p^2)dy + \frac{1}{2}\int_{0.5+\delta}^{\infty} N(0, \sigma_p^2)dy \\ &= \frac{1}{2}Q\left(\frac{\delta + 0.5}{\sigma_p}\right) + \frac{1}{2}Q\left(\frac{\delta - 0.5}{\sigma_p}\right) \end{aligned} \tag{4.9}$$

where $Q(.)$ is the tail probability of the standard normal distribution. Whenever $g(y) = 0$, there is no actual transmission. There are two folds about this scheme, firstly it saves some energy and secondly it reduces the quantization error of the conventional hard decision scheme. The probability of error at the receiver side may be higher but it may offset the quantization error. Besides, from the capacity point of view, this scheme enables higher channel capacity.

Consider the quantized information from the previous section as the input for the SUs to SUs channel. This can be considered as discrete-time ternary input continuous output channel, given by this following channel model.

$$y_s = x_s + n_s \tag{4.10}$$

where $x_s \in -1, 0, 1$ and n_s is an AWGN channel with variance equaling to σ_s^2. The channel capacity is defined as

$$
\begin{aligned}
C &= \arg\max_{p(x)} I(X;Y) \qquad (4.11)\\
I(X;Y) &= \sum_x \int_{-\infty}^{\infty} p(x)p(y|x)log\left(\frac{p(y|x)}{p(y)}\right)\\
&= P_0 \int_{-\infty}^{\infty} p(y|x=-1)log\left(\frac{p(y|x=-1)}{p(y)}\right)\\
&+ P_\epsilon \int_{-\infty}^{\infty} p(y|x=0)log\left(\frac{p(y|x=0)}{p(y)}\right)\\
&+ P_1 \int_{-\infty}^{\infty} p(y|x=+1)log\left(\frac{p(y|x=+1)}{p(y)}\right)
\end{aligned}
$$

where

$$
\begin{aligned}
p(y) &= \sum_x p(x)p(y|x) \qquad (4.12)\\
&= P_0 p(y|x=-1) + P_\epsilon p(y|x=0) + P_1 p(y|x=+1)
\end{aligned}
$$

Let $N(y,x,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-x)^2}{2\sigma^2}}$ represents the conditional probability density function $p(y|x)$.

$$
\begin{aligned}
p(y) &= \sum_x p(x)p(y|x) \qquad (4.13)\\
&= P_0 N(y,-1,\sigma_s) + P_\epsilon N(y,0,\sigma_s) + P_1 N(y,+1,\sigma_s)
\end{aligned}
$$

$$
\begin{aligned}
I(X;Y) &= \sum_{x} \int_{-\infty}^{\infty} p(x)p(y|x)log\left(\frac{p(y|x)}{p(y)}\right) \\
&= P_0 \int_{-\infty}^{\infty} N(y,-1,\sigma_s)log\left(\frac{N(y,-1,\sigma_s)}{p(y)}\right) \\
&+ P_\epsilon \int_{-\infty}^{\infty} N(y,0,\sigma_s)log\left(\frac{N(y,0,\sigma_s)}{p(y)}\right) \\
&+ P_1 \int_{-\infty}^{\infty} N(y,+1,\sigma_s)log\left(\frac{N(y,+1,\sigma_s)}{p(y)}\right)
\end{aligned} \tag{4.14}
$$

$I(X;Y)$ is a function of P and σ_s and above equation can be evaluate numerically. Therefore, we search for P that maximize the mutual information at a given σ_s.

Now, we obtain the capacity by searching for the maximum rate and plot it against SNR of the primary user. Fig. 4.2 shows the rate at different SNR. When P_0 approaches 0.5, the rate approaches 1, which is consistent with the capacity of binary channel. It can be seen that at higher SNR of the primary user, the capacity is also consistent with ternary input, which is approximately 1.58 at P_0 about $\frac{1}{3}$. In both cases, the capacities are maximize when the input are uniformly distributed.

Now, we obtain the capacity by searching for the maximum rate and plot it against SNR, which is shown in Fig. 4.3. In fig. 4.4, we also plot the corresponded P_0 that the capacity is obtained. This P_0 is directly linked to the value of δ and σ_s as we will show in the next section.

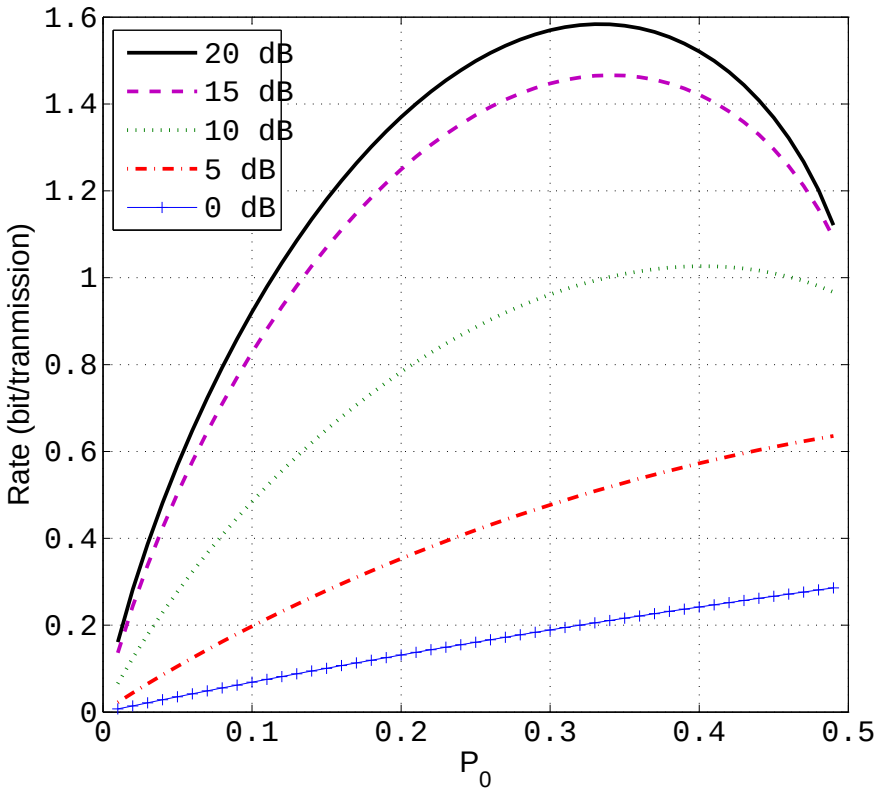


Figure 4.2: Information Rate vs. SNR.

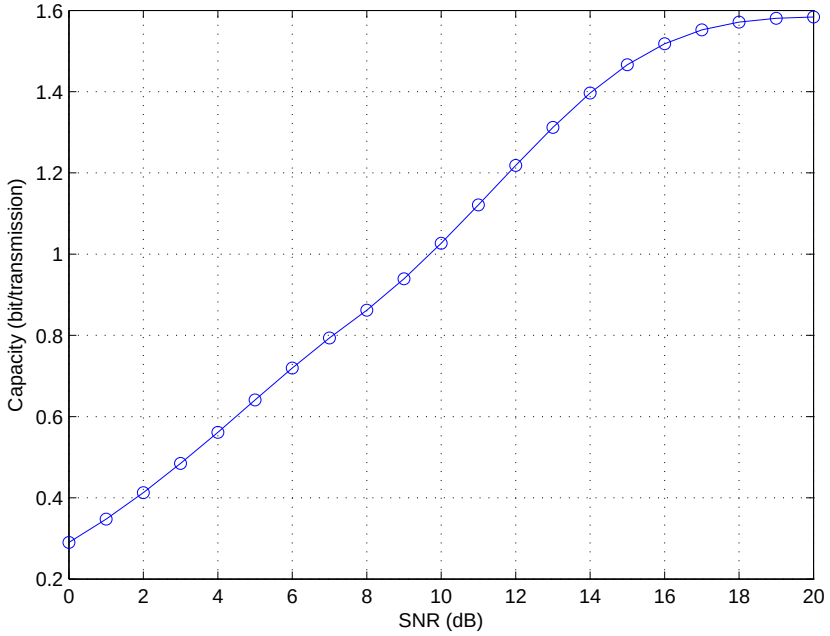


Figure 4.3: Capacity analysis as a function of energy.

4.4.2 Determining Energy Threshold Using Capacity

We know that P (P_0, P_1, P_ϵ) is also a function of δ and σ_p. Hence, knowing optimal P, we can find an optimal δ for a given σ_p by solving inverse function of equation 4.9. Figure 4.5 shows the 3 dimensional plot of the optimal δ at various σ_p and σ_s.

4.5 Simulation Results

Performance evaluations of several hard-decision schemes are shown. There are 4 schemes as shown in table I

Binary mode refers to messages that are quantized to binary or ternary value values -1, 0 and 1 for the ternary case and ± 1 for the binary case. Integer mode refers

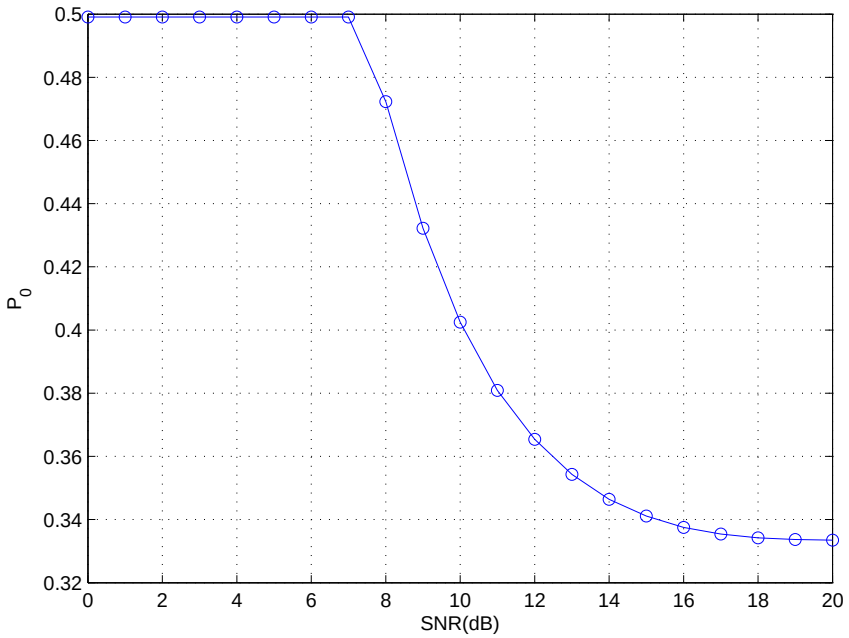


Figure 4.4: Optimal P_0 at which the capacity (maximum information rate) is achieved.

Binary Broadcast	Binary Unicast
Integer Broadcast	Integer Unicast

Table 4.1: Hard decision modes

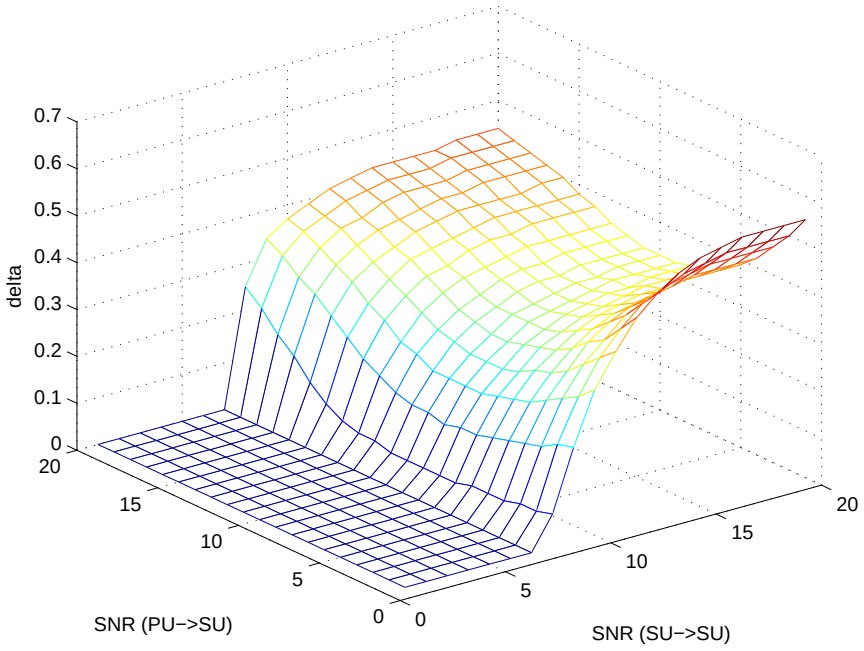


Figure 4.5: Optimal δ at which the capacity is obtained

to messages that initially are binary/ternary values then allowed to be accumulated during the iterations.

Unicast refers to message passing scheme where a dedicated message is sent from node i to j, while broadcast scheme broadcasts the same message to all of its neighbors. For a fair comparison, we normalize the energy usage of both ternary-input and binary input scheme. For simplicity, we simulate this in a network of 36 nodes arranged in a grid-like topology.

The performance comparison are plotted in fig. 4.6 and 4.7. Fig. 4.6 shows the performance against iterations at SNR equaling to -3 dB. Fig. 4.7 represents performance when the SNR is varying between -4 dB to 4 dB.

The results show that the ternary-input schemes out perform the conventional schemes in all scenarios about 2-3 dB gain at the probability of 10^{-3}. The "Integer-Unicast" scheme performs better because no truncation is applied while the "Binary-Unicast" is slightly worse, which is of course suffering from the truncated information. The broadcasting case which is a relaxed version where the messages are the same for all neighbours, performs a little bit worse; but with the advantage of the broadcasting nature of wireless signal and the less complexity, this scheme can be an alternative to the best one.

4.6 Conclusion

Our contributions in this paper are, firstly, the proposed distributed algorithm for spectrum sensing, which is based on the combination of an efficient belief propagation algorithm with the use of hard information instead of soft information resulting in

an effective scheme that does not require centralized unit to combine the sensed data yet it is not energy burden as we use hard information as well as applying a threshold that does not allow unreliable messages to be propagated because of energy inefficient (reliability/energy unit).

Secondly, we provide a systematic way to calculate the threshold δ by modelling the channel among sensing nodes as ternary-input continuous-output AWGN channel and finding δ in which the capacity is achieved.

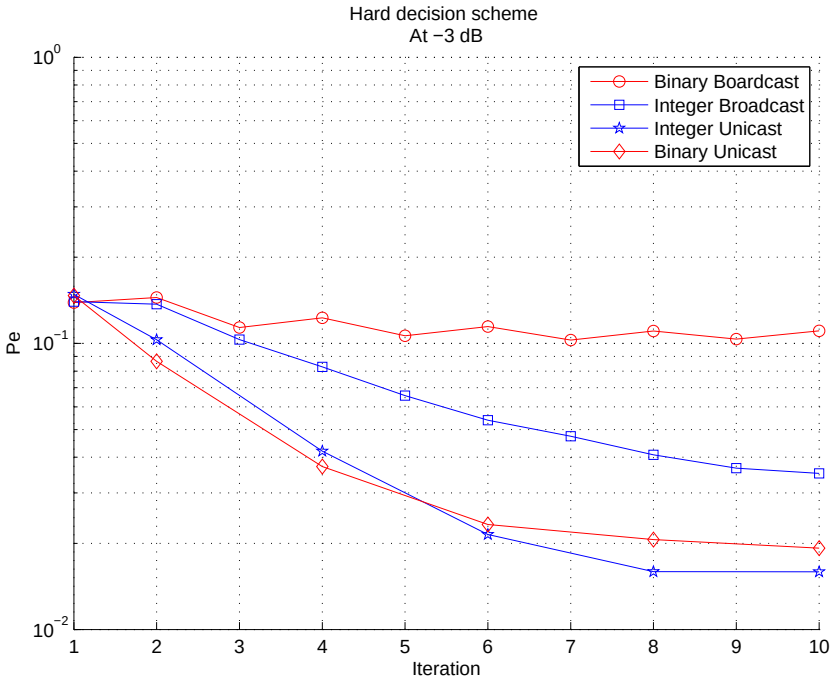


Figure 4.6: Performance of various hard-decision schemes at 3 dB

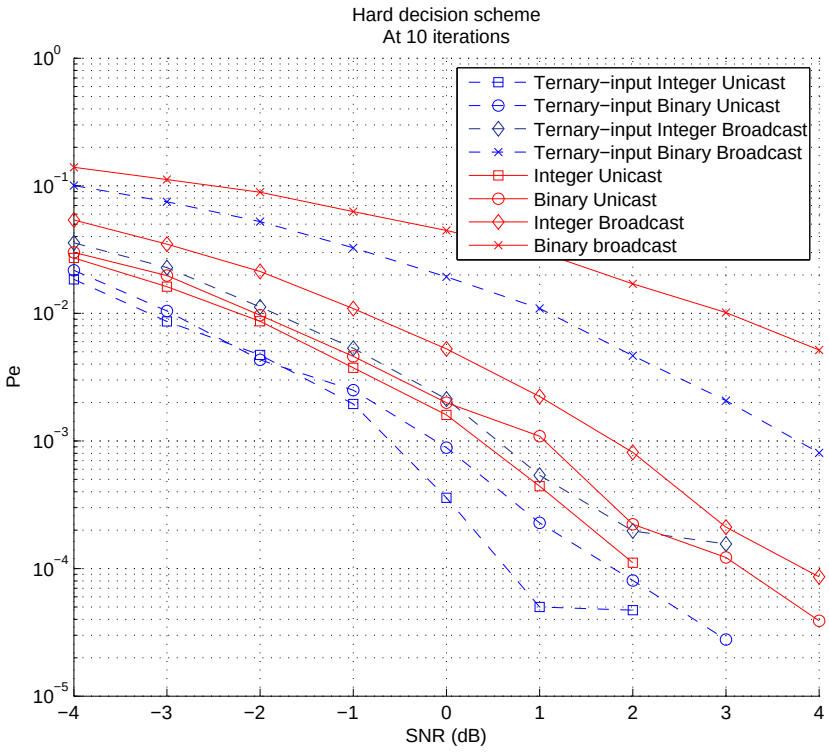


Figure 4.7: Performance of various hard-decision schemes at 10 iterations

Chapter 5

Network Localization on Unit Disk Graphs

5.1 Introduction

Localization using GPS is a mature technology that has been deployed and proven very useful in a myriad of applications in military, commercial and personal uses. In other scenarios, like wireless sensor networks where information about locations is as important as measuring data itself, GPS-equipped devices may not always be practical, since the sensing devices are often limited by the cost, power or lack of the line-of-sight from the satellites.

This paper considers a general scenario involving a network system, where some *anchor* nodes already know their exact locations (through GPS or other methods), while the rest (*non-anchor* nodes) do not and await to be localized. While the GPS

system can be viewed as a single-hop method, network localization is usually classified as multi-hop localization [37]. Distance information between nodes are often attainable by measuring wireless modalities, such as time-of-arrival (TOA), angle-of-arrival(AOA), and received-signal-strength (RSS) [37].

In general, one may assume that no non-anchors have distance measurements from 3 or more different anchors (anchors not in the same line), because otherwise these non-anchors can be quickly localized through geometric trilateration or triangulation (single-hop method) and become anchors.

Since a non-anchor does not have direct reachability (distance information) to enough anchors to be localized by itself, they must share resources and cooperate to accomplish the task together. Many researchers formulate this multi-hop localization problem as some type of optimization problem (non-linear, convex, semi-definite-programming) where standard algorithms can be applied, usually in a centralized manner [42, 43]. Other approaches fall into the heuristic category where the distances from a non-anchor to anchors are approximated using different techniques, varying from the number of hops [44] to the sum of the exact distances [45].

In addition to algorithmic study, there is also on-going effort to lay down the theoretical building blocks of this problem, and many useful results are being reported. In the theoretical framework, the network is commonly modelled as a geometric graph, where distances are exact and represented by straight-line edges (with definite lengths) between nodes. (More system models will be discussed in the next section.) Part of the theory deals with the properties/conditions that lead to unique localization solution.

For instance, rigidity[1] of the graph guarantees the existence of the unique realization of the graph. If one can check for the rigidity purely based on the graph connectivity (e.g. 3-connected, 6-connected) [39], then one can immediately determine the existence of the solution without even trying to realize the graph, and there exist efficient algorithms to verify rigidity and the related conditions. Despite such existence tests, however, realizing a rigid graph remains a challenging problem in general.

One exception is *trilateration graphs* for which efficient realization algorithms exist. A trilateration graph is one such that, at any time, at least one non-anchor is connected to enough anchors (at least 3 for 2D) to determine its location and becomes an anchor, and this condition holds iteratively through out the network. For an arbitrary geometric graph model (non-trilateration graphs), in term of the complexity to realize the graph, they are NP-Hard [40, 41].

In an attempt to find more effective way to tackle the problem, unit-disk-graph model is also used to present a simplified version of the arbitrary graph, in the hope to reduce the problem to the point where efficient algorithm can be applied. A well-known network model widely used for modelling wireless ad-hoc networks, a unit disk graph preserves the constraint that two nodes are connected if and only if they are within r distance away.

Variations can also be made to suit one's need, such as uniform or non-uniform radius r across the network, but the key concept is that r defines a clear-cut boundary of the connectivity. Although network localization can be modelled this way, and may even be further simplified to aligning nodes in the integer domain, the problem

[1] In this paper, when referring to rigidity, we mean global rigidity.

remains NP-Hard [41].

On the other hand, however, NP-Hard is considered from a worst-case perspective. For example, in a non-trilateration graph, if a large number of non-anchors are assisted by only 3 anchors, then NP-Hard stands strong for its complexity. In a practical applications, there may be other conditions and parameters (such as denser networks, larger communication ranges and more anchors), which, although do not reduce the theoretical complexity from NP-Hard to something easier, do help improve the average complexity as we will later show through our proposed search algorithm.

In this paper, based on the unit-disk-graph model, we re-evaluate the relation between graph rigidity and unique positioning, and study practical search algorithms for network localization. We point out that the implicit properties of the unit disk graph can provide great benefits in localization. Specifically, while "being connected" bears good information on the separation and possible posts of the nodes, "not being connected" contains equally important information about their locality in terms of where they cannot be. However, this latter information has been largely ignored. We show that an active exploitation of the forbidden region (as well as the the feasible region) can greatly change the landscape of the problem and significantly expedite the practical search algorithms.

We also show that the conventional rigidity requirement is no longer the necessary condition for the uniqueness of the solution. This means that rigid graphs become only a subset of the total set of networks where a unique solution can be found. Exercising on a grid graph with integer coordinates, we further propose a tree-search algorithm that can find the solution within a manageable complexity.

The proposed searching algorithm is derived from depth-first search, but makes

essential use of the multiple conditions and constraints inferred by the unit disk graph, to actively rule out invalid branches and keep the intermediate results consistent. While the problem is still NP-hard, simulations show that the search complexity is significantly lower than if the "no-connection" information is simply ignored.

5.2 Preliminary Definitions

In this section, we put formal definitions on which our problem is formulated and based. We start with geometric graph, which is the extension of definitions of general graphs $G(V, E)$. Note that we will omit $G(V, E)$ as it is simple and broadly understood.

Definition: Geometric Graph 5.2.1. *For a geometric graph $G(E, V)$, there is value associated to a node v, which is a point in 2D space, e.g. (x, y). Then we also attach the distance d_{ij} to the edge e_{ij} connecting one node i to anther j. The relationship between distances and locations is straight forward.*

$$d_{ij}^2 = (x(v_i) - x(v_j))^2 + (y(v_i) - y(v_j))^2 \tag{5.1}$$

Note that the geometric graphs we refer to are defined in two-dimensional space. We can further define a unit disk graph as an geometric graph where e_{ij} exists only if $d_{ij} \leq r$. A uniform random graph is a graph where the coordinate of a node v is uniform random $[0, 1]$ both x and y of a node v. Then a uniform random unit disk graph is the combination of both definitions. The rigidity of a graph can be defined as the following.

Definition: Graph Rigidity 5.2.1. *For a geometric graph $G(E,V)$, let $G_c(E_c,V)$ be the complete graph of G. Let $E_{c'} = E_c - E$ be the edges in G_c that is not in the original graph G. Let $d(e_{ij})$ be the distance or length of the edge e_{ij}.*

A graph $G(E,V)$ is not globally rigid if there is another graph G^1 such that $d(e_{ij}) = d(e^1_{ij})$ for all $e_{ij} \in E$ and $e^1_{ij} \in E^1$, but there exists $d(e_{ij}) \neq d(e^1_{ij})$ for any $e_{ij} \in E_{c'}$ and $e_{ij} \in E^1_{c'}$.

A descriptive and informal concept of graph rigidity can be explained as follows: Replace vertices with free-moving joints and edges with rigid string; then rigidity means that the shape of the graph does not change when applying a vector of forces to any subset of the joints. Figure 5.1 depicts an example of a rigid graph and non-rigid graph.

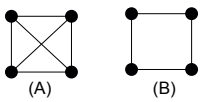


Figure 5.1: (A) Rigid Graph and (B) Non-Rigid Graph.

5.3 Problem Formulation and Rigidity Condition

We formulate the localization problem as a unit disk graph $G = \{V, E\}$ consisting of $N = |V|$ vertices (nodes) residing in a 2-dimensional (2D) plane. A pair of nodes (i, j) whose distance falls within a uniform radius distance of $r = \{r|r \leq 1\}$ is assigned an edge (link) $E_{(i,j)}$ with distance parameter $d_{(i,j)}$. We assume the graph is connected and the vertices are placed uniformly at random in the area. We also assume that a special set of M know their absolute locations, where $3 \geq M < N$ and these M nodes do not lie in the same line, which is the necessary condition to avoid ambiguity from

simple transformation like rotating, flipping, or shifting the graph [38]. A realization $p(V_i)$ of G is a point assignment of a node V_i into the plane, and is specified by a coordinate vector x_i.

For simplicity, we further constrain x_i to be aligned at integer coordinates only and use the Euclidean distance square $d^2_{(i,j)}$ as the distance metric. Such a simplification transforms the graph to a grid graph that deals with integer-valued distances only, but does not lose the generality of a random graph (since any real value with a given level of precision can be scaled up to an integer, with the rest of the graph also scaled proportionally).

Using this model, the network localization problem can in general be stated as follows: Given a graph defined above, find the absolute/unique location of each node:

$$\begin{aligned} & \text{find} : x_i, && \forall i \in 1..N \\ \text{s.t.} \quad & g(x_i, x_j) = d^2_{(i,j)}, && \forall E_{(i,j)} \in E \\ & x_i \in \mathbb{Z}^2. \\ & x_i \neq x_j, && \forall i, j \end{aligned} \tag{5.2}$$

where $g(x_i, x_j) = ||x_i - x_j||^2$ is the distance metric. This problem formulation can be classified as a non-linear integer optimization problem, and has an NP complexity that is as difficult as any non-linear integer optimization problem [41].

Now consider the implication of the unit disk graph. If two nodes do not have connection, it implies that these nodes must be away from each other by more than r. This condition can and should be explicitly put in the mathematical formulation,

to restrict the search space and minimize the search effort:

$$\begin{aligned} & \text{find}: x_i. && \forall i \in 1..N \\ \text{s.t.} \quad & g(x_i, x_j) = d^2_{(i,j)}, && \forall E_{(i,j)} \in E \\ & x_i \in \mathbb{Z}^2, && \\ & x_i \neq x_j, && \forall i, j \\ & g(x_i, x_j) > r^2, && \forall E_{(i,j)} \notin E. \end{aligned} \tag{5.3}$$

As we mentioned earlier, there is an equivalence relation between global rigidity and unique realization of a graph. For basic background and more detail of graph rigidity, we refer readers to [38].

Now that we have used the unit-disk-graph model, if we strict ourselves to only the conditions in (5.2), then global rigidity and unique realization still holds. However, when we follow the constraints in (5.3), then a graph with unique realization no longer implies rigidity. It is enough to confirm this finding through a counter-example. Figure 5.2 shows a graph where we can find a unique localization solution without requiring rigidity. Given 3 anchor nodes (solid black) and 5 edges, the non-anchor node (gray) has 2 possible locations under conditions in (5.2).

However, with the constraints in (5.3), only one graph realization (A) is valid, and the other possibility can be ruled out since the non-anchor falls within the transmission range of an anchor (and should otherwise has an edge to it).

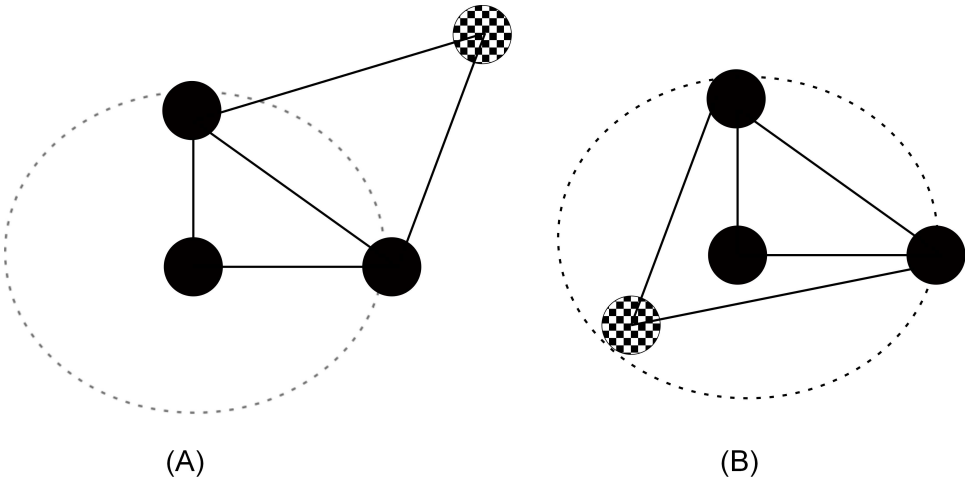


Figure 5.2: (A) Valid solution; (B) Invalid solution, under disk-graph assumption.

5.4 Tree-Search Algorithm

5.4.1 The Algorithm

The proposed tree-search localization algorithm is based on the idea of examining the realization possibilities constructively, similarly in flavor to that of list-decoding in channel codes. The anchor nodes form the initial set of realized nodes R, and all the non-anchors belong to the un-realized set $U = V \backslash R$. These two sets are collectively viewed as an intermediate instance T_k^l for constructing a complete realization, which is achieved when U is empty and all nodes are in R.

Our approach is to efficiently go through all possible instances of the realization and pick the right solution out of that pool. By setting the initial instance T_0^0 as a root node of a tree structure, we expand this instance to new instances by moving a non-anchor from the un-realized set to the realized set R, at all the possible locations that this non-anchor can possibly be. Each possible case corresponds to a new instance T_k^l, where l is the level of the tree and $k = 0, 1, 2 \cdots$ is just an index in that level.

The general process of growing the tree is to pick a non-anchor $j \in U$ that has connection to at least one realized node $i \in R$. The distance information $d_{(i,j)}^2$ states that node j must sit in this radius away from node i, and since location must be aligned in integer coordinates, the possibilities are discrete and countable (and can therefore be indexed by $k = 0, 1, \cdots$). For efficiency, many of these possibilities should be immediately trimmed out by testing the connectivity/distance conditions between node j and all the other nodes in R. Violation of any edge condition or any no-edge condition should leads to the removal of this instance.

Clearly, each time a new node is added to R makes the tree to grow one more

level. The process continues until all the $(N-M)$ levels are completed, and the survival instances at the $(N-M)^{th}$ level are the solutions. We declare all the nodes to be unequivocally localized if there is only one survival. Since all the realization possibilities are either examined or ruled out due to connectivity violations, the tree-search algorithm is optimal.

It may appear that this tree structure would grow exponentially, with all the possibilities listed at any given level. However, this worst case scenario does not really happen, because we make active consistency check using the conditions in (5.3). With the conventional network localization formulation in (5.2), a sparse graph (i.e. one with relatively small reachability r) is usually most time-consuming to localize, due to the very limited edge information. The proposed strategy as formulated in (5.3) is particularly helpful for such cases, because it makes effective use of the lots of no-edge constraints, and therefore drastically reduces the search instances!

Algorithm 1 Constructing Tree

```
T_0^0 ← {R_0^0, U_0^0}
for (l = 1, l ≤ (N−M), l++) do
  Pick node n from U_k^{l−1}, n is a neighbor of R_k^{l−1}, ∃k
  for all T_k^{l−1} ∈ Tree do
    X = sub locations(n,T_k^{l−1})
    for all x ∈ X do
      add T_x^l as a child of T_k^{l−1}
    end for
  end for
end for
```

Claim 1. *Algorithm 1 produces at least one valid solution at the leaf node $T_k^{(N-M)}$, if the graph is realizable.*

Proof. We show this through mathematical induction. Starting from level 0, the

Algorithm 2 Sub Locations(n,T)

```
.{find all valid placements of a node n}
. {Return: A point vector Y = {y_1, y_2, ...}}
R ← realization in T
m ← neighbor(n),   m ∈ R
d ← distance E_(m,n)
X ← all points of radius d at m
for all x_i ∈ X do
   temporary place x_i in R
   if validate(x_i,R) using eqn (5.3) then
      add x_i to Y
   end if
end for
return Y
```

instance T_k^0 has M nodes (anchors) in the realized set R whose positions are valid. Note that "being realized" means that a tentative locality has been assigned to that node, but the assignment may be invalid (incorrectly realized) since violations have not shown.

Based on the assumption of connected graph, at level 1, the tree constructed from the instance from the previous level must contain at least one instance whose $(M+1)$ nodes in R are all being *correctly* realized. Now assume at any level l, where $1 \leq l < (N-M)$, there exists at least one instance T_k^l that that has $(M+l)$ correct locations. Since the tree at level l+1 is produced from all possible placements of a node to the instances from level l, then among all the child instances that are inherited from the correct instance at level l, there must be at least 1 correct instance whose $(M+l+1)$ nodes in R are all correctly realized. □

Claim 2. *Algorithm 1 is optimal, that is, it produces all the valid solutions at the leaf node* $T_k^{(N-M)}$.

Proof. From the algorithm, when the tree expands from level l to l+1, all the possible instances are included (exhaustively). A subtree is trimmed off prematurely if and only if the realized set at that level is already violating some of the geometric distance conditions. Hence, no single valid solution may be left out from this search algorithm. □

After constructing a complete tree, we can instantly obtain the solution by looking at the instances at the leaf level. If the graph is uniquely realizable, then only one instance at the leaf level.

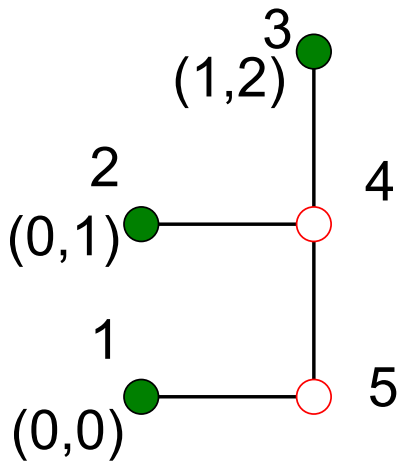


Figure 5.3: An example of a connected network with 3 anchors (solid green) and 2 non-anchors(hollow red).

5.4.2 Example

Consider a connected network in Fig. 5.3. Node 1, 2 and 3 are anchors, while 4 and 5 are non-anchors. Suppose radius $r = 1$ and nodes are aligned in integer coordinates. We start the initial realization instance with all the anchors and keep the unknown set aside, resulting in instance T_0^0 in Fig. 5.5(A). Then at level 1, we pick a node, say,

node 4, which is connected to the realized sub-tree. Note that node 4 is connected to node 2 and 3 and two possible locations are shown in Fig. 5.5(B).

The invalid instances are omitted to save space. At the next level, we pick node 5 and try to realize it in all the different locations possible. All, except for one, of the tentative realizations will be eliminated due to violations of some rule (distance rule, non-reachability rule, two nodes cannot collide). The survival is the (only) valid realization.

5.4.3 Implementation, Memory and Complexity

The above example illustrates how the tree is constructed, and how invalid sub-trees are eliminated as soon as possible. Nevertheless, implementing this algorithm for large networks (lots of non-anchors) can consume a huge amount of memory. To overcome this practicality, instead of width-first search, the *depth-first* search can be used along with the dynamic creation of the tree. The maximum space requirement is then limited to $O(N)$.

The computation complexity is $O(|T|)$, where $|T|$ is the size of the resultant tree, which is directly related to the specific network topology and properties (such as diameter, average node degree, and communication radius), as well as the order in which the non-anchor nodes are being selected to realize.

At level k, suppose a non-anchor j, which connects to some realized node i with distance $d_{(i,j)}$, is chosen to be realized. Then the spanning factor of the tree at this level is upper limited by $D(d_{(i,j)})$, where $D(d_{(i,j)})$ is the number of integer coordinates on the circle of radius $d_{(i,j)}$. Note that many of these instances may be immediately eliminated after we evaluate all the other rules (formulated in (5.3)). How much or

what portion of these tentative realizations can be immediately discarded is different from case to case and very hard to quantify, but at least we should make $D(d_{(i,j)})$ as small as possible, especially for early levels of k.

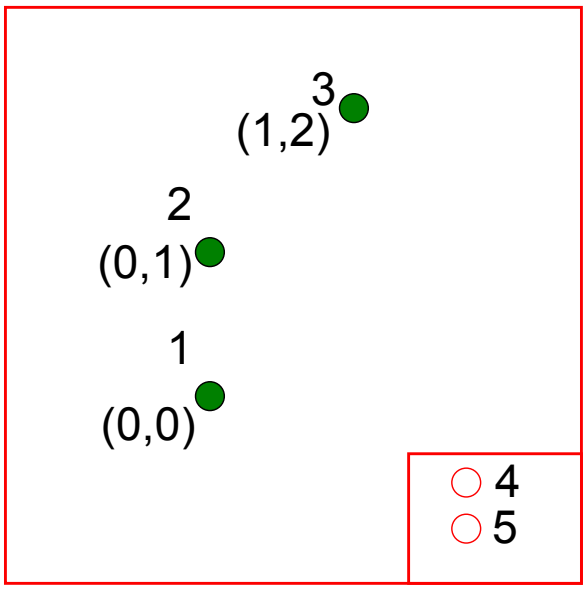
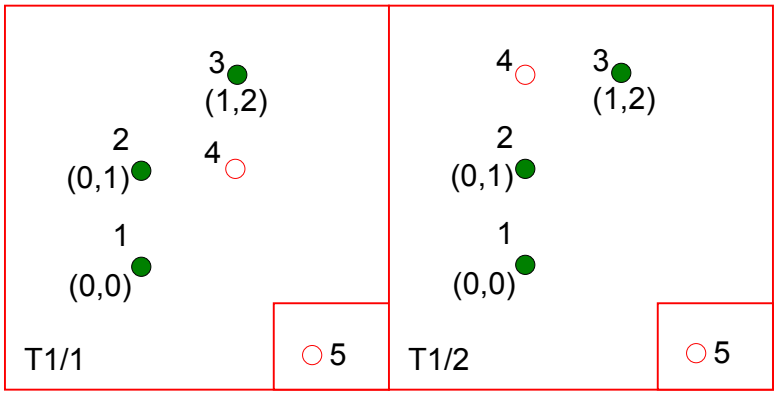


Figure 5.4: (A) Level 0.

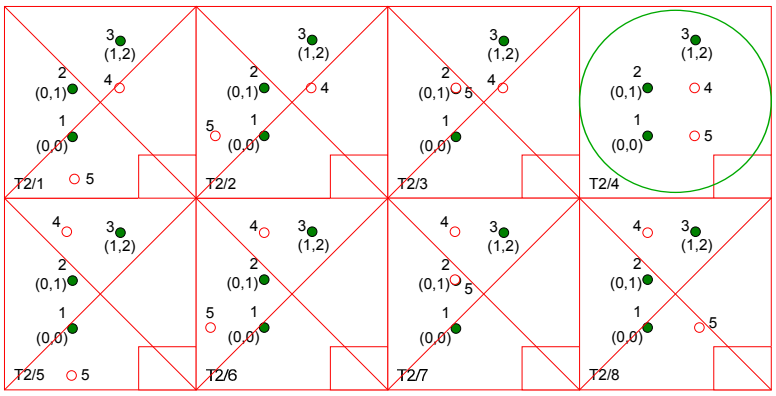
5.4.4 Implementation improvements

This section dedicates to the algorithmic/strategy improvements that further reduces the complexity. There are several opportunity that we can heuristically exploit, for examples:

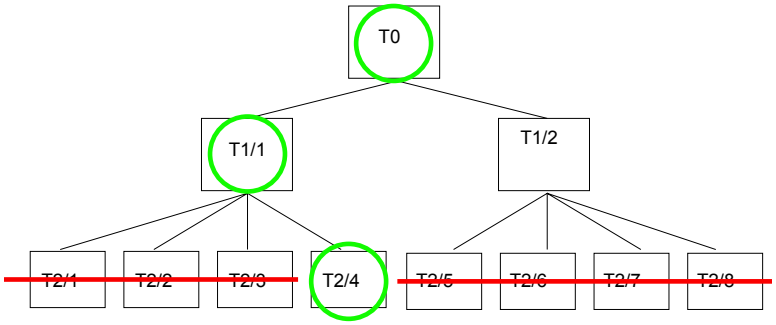
- parallel implementation: this itself can reduce the computation time by a linear factor of the additional processing units. However we are looking at the better parallel gain, for example information sharing of the newly discovered nodes among threads globally reduce the complexity of each processing unit.



(A) Level 0.



(C) Level 2.



(D) The overall tree T.

Figure 5.5: Example.

- tree expanding strategies: As of the basic algorithm proposed in alg.1, the new node is chosen randomly from any node that is connected to the intermediate-realized instance. We can wisely choose a node based on the distances, hoping that reaching the other ends of the network faster will help us eliminated the unwanted paths.

- It is also promising to consider picking a new node that has more connections to the intermediate graph. In the best case we are exploiting the tri-lateration (a node has 3 or more connections to the known nodes) of a graph where only one possible location is determined directly. Even though it is not tri-lateration we should get at least 1 connectivity and expecting 2-connectivity as well. So this scheme in any case, will perform better than the random choosing.

We show the simulation result in the later section to show that these modifications are really helpful.

5.4.5 Noisy distance measurements

Noisy measurements compromise several joyful advantageous, such as the early discretely dismissal of the traversal paths. In an effort to bring our algorithm out of the ideal settings and to face real world effects, we propose a modification/relaxation that would lead our algorithm to be more flexible and robust to the presence of noise, while keeping the efficiency at the some certain acceptable ground.

We move a step closer to a possible realistic environment. First if we consider the unit disk graph model, the communication range is idealized by the cutting off value r. In reality communication range is infinity, of course with the intensity fading away

rapidly. Unit disk graph with range r can be viewed as the representation of the case where we can intercept strong signal. To capture this characteristic, it should be reasonable to define multi-range disk graph model. We demonstrate this in the case of 2 radius model, inner radius r_1 and outer radius r_2.

We define the model as follows: Any two nodes inside the range r_1 is said to be strong and reliable link, e.g. it is 100% that the two establish a connection. We then define the region between r_1 and r_2 as moderate link where two node within this range can establish connection with some probability $p_r 2$. Finally, we treat the region outside r_2 as poor signal with 0 probability. Ones can directly extend this to multiple level, but we will not go into that detail. Figure 5.6 demonstrates the idea. Between node A and B, the two are within r_1 region, so they establish the connection. For region r_2, for example, node B and node C, it can be a link between them with property of 50 percent.

Then what does the this mean to our problem and solution? It implies that when there is no link between two nodes in the r_2 region, it does not means that we can discard the possibility that the other node may be in this range. So this model can be viewed as the union of the standard localization algorithm and the algorithm making use of no-link information. The problem formulation, shown in eqn. (**??**) is similar to the previous stated,except that the last constraint is only enforced on the inner region r_1.

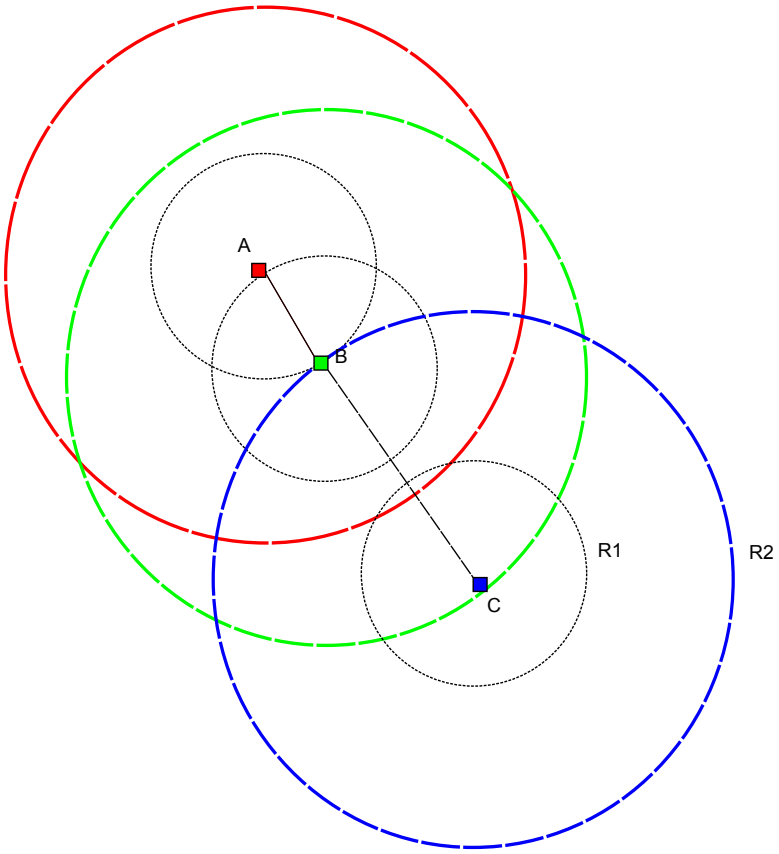


Figure 5.6: An example of inner radius line and outer radius link

$$\begin{aligned} & \text{find} : x_i. && \forall i \in 1..N \\ \text{s.t.} \quad & g(x_i, x_j) = d^2_{(i,j)}, && \forall E_{(i,j)} \in E \\ & x_i \in \mathbb{Z}^2, && \\ & x_i \neq x_j, && \forall i, j \\ & g(x_i, x_j) > r_1^2, && \forall E_{(i,j)} \notin E. \end{aligned} \tag{5.4}$$

In the next section, we will compare this multi-range model to the uniform unit disk graph.

5.5 Simulation Results

In our simulation setup, a set of N nodes are randomly placed in a square grid of $C \times C$. The connectivity is established by radius r. A small portion of nodes (M) are randomly picked as anchors. Fig. 5.7 shows one example of a generated instance.

For a given set of parameters (C, r, N, M), we collect the average traversal times (traversal time is equivalent to total number of instances or total number of branches being visited), averaged over many random network realizations using the given parameters.

In general, a small percentage of anchors makes network localization strenuous. This is particularly so in the conventional case formulated in (5.2). By actively exploiting "useful implicit distance information," not only are complexity and search time drastically reduced, but the grave impact caused by sparse anchors is also drastically relieved. Fig. 5.8 compares the traversal time by using rules in (5.2) (dashed

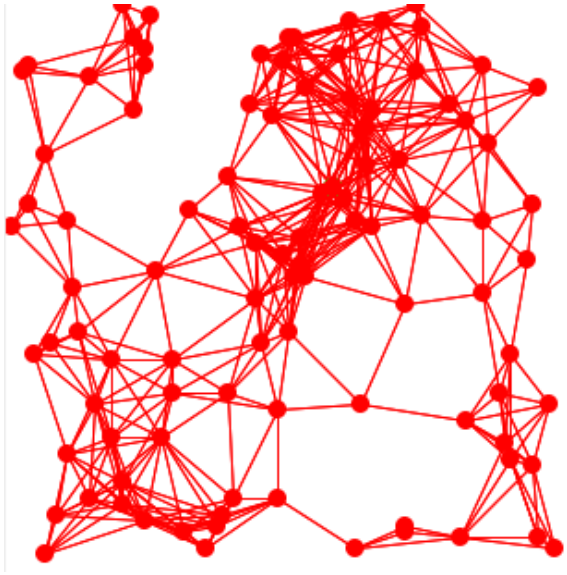


Figure 5.7: A random network of 100 nodes in a 100×100 square grid with radius $r = 20$

line) and rules in (5.3) (solid line). We also perform two different search strategies, one is a random picking of an unknown node when expanding and the other is picking an unknown node that has the most connections to the intermediate graph.

The experiments are performed on random graphs with $C = 100$, $r = 25$, $N = 100$, and varying M. We see a large magnitude of reduction in traversal time due to (5.3), especially when only a very small percentage of anchors present. For example, with 1% of anchors, the traversal time reduces from 252 to a merely 2.57 per non-anchor, a speed-up of some 100 times in random picking of a new node and about 2 times using the most-connected strategy.

In addition to the impact of the percentage of anchors, we also evaluate the impact of node density, or, equivalently, the communication radius r. Fig. 5.9 shows the case using rules in (5.2) and (5.3). Similar observations can be made. In the conventional case, a sparse network (small communication range and low node degrees) makes

localization very difficult. Again, our new problem formulation and search strategy can significantly reduce this negative impact, more than 5 times reduction from 379 to 61 traversal time/node when the radius is 0.2, using the random algorithm while the most-connected algorithm is about 0.5 times better.

However, when the network is sufficiently dense (e.g. $r = 0.3$), we see that the additional information is no longer the advantage because the larger range implies that a node can reach the anchors easier within a few hops or even in a single hop. In all cases, the results show that the unit disk graph model with priori information is useful when other information such as the connectivity and anchor nodes are limited.

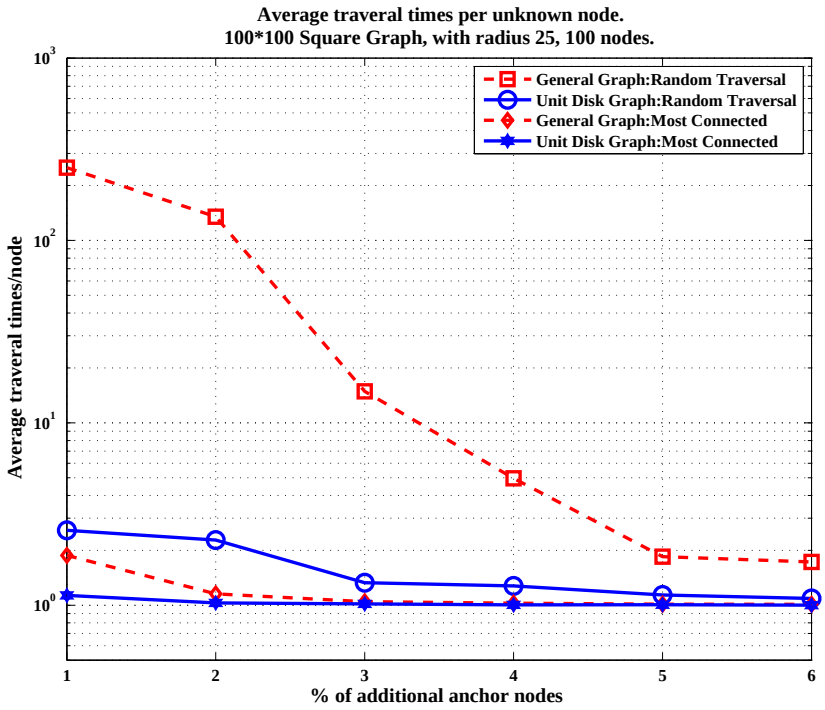


Figure 5.8: Comparison between using (solid line) and not using (dashed line) the no-edge information, for varying percentage of anchors.

In the second model where we classify the signal reliability into different level.

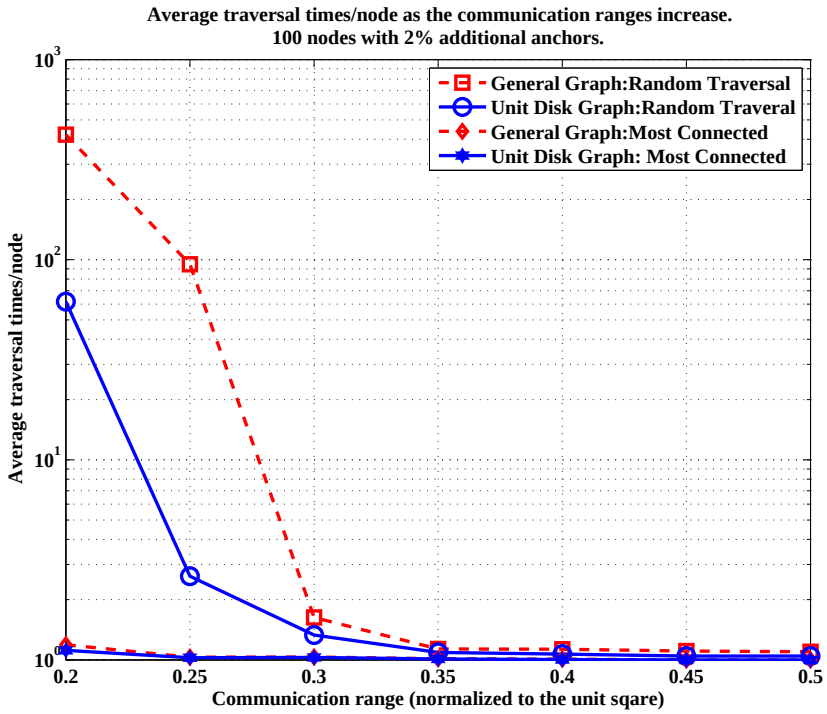


Figure 5.9: Average traversal time as the communication radius r (i.e. density) increases.

It gives more granularity to our model and should offer a better performance. We simulate this model here with $r_2 = 2r_1$, that is the outer radius is 25% larger than the inner radius. And the probability that the region $(r_1, r_2]$ will establish the connection is 0.5. Figure 5.10 compares 2 approaches, the enhanced algorithm utilizing non-connectivity constraint and the latest method, multi-range scheme. The multi-range scheme out performs former scheme as it has more useful constraints to be used.

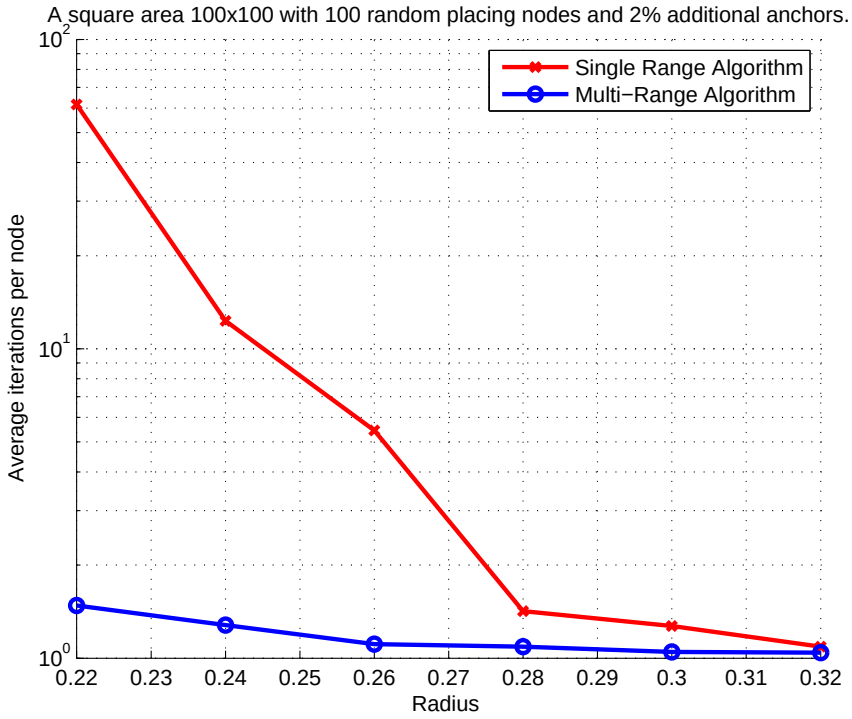


Figure 5.10: Average traversal time as the communication radius r (i.e. density) increases.

5.6 Complexity Analysis

In this section we evaluate the average iterations in the random unit disk graph model. The simulation suggest that the complexity is in the manageable area, with a

reasonable network size. We affirm this result by approximating the average iterations by using the probabilistic property of the random network. The random network nodes distribution is often modeled as Poison process. We show this along with an example as the followings.

Assuming we have $n = 100$ nodes in a $s^2 = 100 * 100$ square. A chance that a node with communication range r has it k-1 neighbors (equivalent to having k nodes including self), is model as Poisson process with parameter $\lambda = \pi r^2 \frac{n}{s^2}$, where Poisson process is defined as

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{5.5}$$

We would like to approximate the average traversal time per node in order to solve for the locations. The best case is that we use only 1 computation unit per note. During the process of trial-error placing a new node. Assuming that all neigboring nodes of this new nodes know their locations and graph is connected, then we have 3 cases.

- this new node has 1 neighbor (k=2), the worst case is this node has to exam 8 possibilities.
- 2 neighbors (k=3). In this case, only 2 possible locations.
- 3 neighbors (k=4). The location is fixed.

The average number of trials before locating a node can be calculated as the following:

Let k be a random variable representing the number of nodes counted in the radius of r. k follows Poisson distribution.

$$E[\text{number of trials}] = \frac{8 * P(k=2) + 2 * P(k=3) + 1 * P(k=4)}{1 - (P(k=0) + P(k=1)))} \tag{5.6}$$

Then we plot this bound and compare to the simulation result. It serves as the lower bound since we have assumed that the neighbors are known. Figure 5.11 shows the plot of this analysis a long with the simulation result (line Analysis 1).

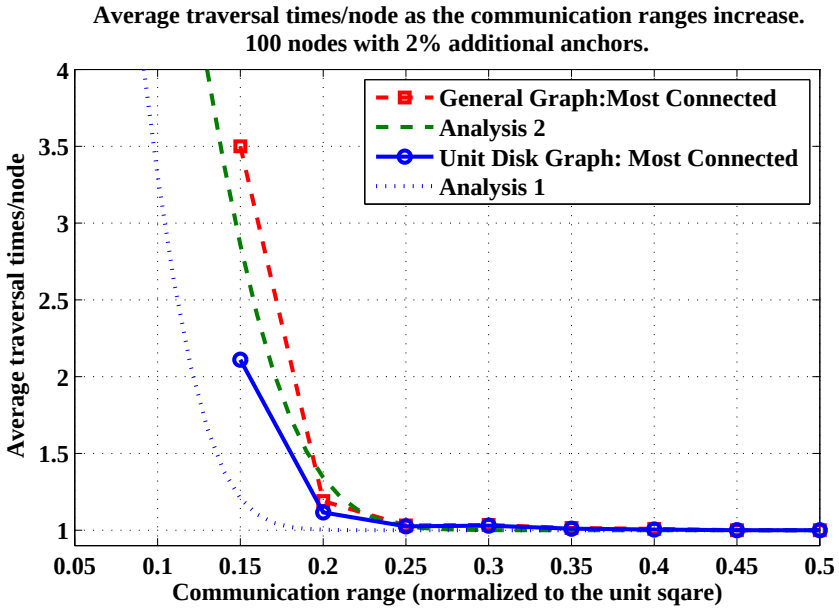


Figure 5.11: Analytical bound versus the simulation results.

To make it to be further more realistic, we assume that with equal probability, a neighbor node could know or not know its location. Then the number of unknown neigbors can be 1,2,3, ... given the number of neighbors is known. This follows binomial distribution and can formally be written as:

$$p(k\text{unknown}|n\text{neighbors}) = \binom{n}{k} p^k (1-p)^{n-k} \tag{5.7}$$

Of course, one of the neighbors must know its location because it is the one that this node of interest is expanded from. Then we can modify our previous bound to make it be more realistic on the average of the traversal time. Let k_n be the number of known nodes.

$$E[\text{number of trials}] = \frac{\begin{array}{l} 8 * P(k=2) \\ +8 * P(k_n=2|k=3)P(k=3) \\ +2 * P(k_n=2|k=3)P(k=3) \\ +8 * P(k_n=2|k=4)P(k=4) \\ +2 * P(k_n=3|k=4)P(k=4) \\ +1 * P(k_n=4|k=4)P(k=4) \end{array}}{1-(P(k=0)+P(k=1)))} \tag{5.8}$$

Plotting this up results in a closer approximation to the simulation result (the line Analysis 2), as also depicted in fig. 5.11.

5.7 Conclusion

Unit disk graphs are a popular wireless network model, in which the transmission range r sets a clear cut on node connectivity (and hence distance information). Extracting useful features of this model, we have re-formulated the conventional network localization problem as one that actively exploits the "no-edge" information as well as the edge information. We show that the rigidity condition can then be safely relaxed, and still result in unique graph realization.

We further developed a (depth-first) tree-search algorithm with active branch-trimming, which runs efficiently in real world setting as the simulation results confirm. We conclude the paper by emphasizing that that information regarding two nodes not being connected can be as important as knowing that they are connected! This implicit information significantly reduce the average complexity.

Chapter 6

Conclusion

Our research interests fall into the area of distributed and cooperative computing and algorithm in networked system, especially in wireless networks. We focus on 3 specific applications, cooperative data communication, cooperative spectrum sensing for cognitive radio and the localization in wireless network. The contributions on each application can be summarized as follows.

In the first application, we proposed the distributed receiver cooperation for single source to multiple receivers broadcasting. This can be considered as one of the so called system-on-graph where the underlying graphical model is built dynamically on-the-fly. There are 3 steps in this scheme. First the source broadcasts packets to the receivers. Second, the receivers relay packets to each other. Finally each receiver construct the LDPC code and decode the data packets.

The scheme employs the binary random network coding which produces good LDPC codes. We also proposed quasi-cyclic offset vector shifting that further improves the coding gain. Quasi-cyclic shifting will eliminate the limitation when we

have a small number of receivers or packets. We proposed a scheduling scheme for data relaying phase that will avoid collision when there is no centralized control unit. The simulation results show that outperforms the case of no cooperation.

In the second application, spectrum sensing for cognitive radio, we looked at the case of a group of secondary users trying to make the decision distributively and cooperatively. We model this as a Markov Random Field problem, a class of the probabilistic inference problems. Then we apply the message passing algorithm, whereas both communication protocol and computation method are embedded in the algorithm. The problem is formulated nicely in this model and we later derived it to simple message passing and combining rules.

It operate on the connectivity graph model, which is consider another form of the same concept of the system-on-graph. There are simple steps in each iteration as follows. First each unit initialized it believe value and the initial message with the log-likelihood ratio of the measurement. Then a message prepared for a neighbor node is the summation of all the of the messages from previous iteration, except that neighbor. After some iterations, each node can make decision based on the summation of all messages in the last iteration. Through simulation, the schemes produces a comparable result compared to the centralized scheme.

If the graph is tree-like, then our method produce exact result as the centralized system. This is where we added a pre-network trimming to trim down any random network into tree-like network using a well-known distributed spanning tree algorithm.

Our analysis consists of two part. First we look at the performance bound of scheme. By using the ordered statistics, we can derived the mean-square error bounds.

In the second part, we analize the behavior of our algorithm, in term of the convergence and the relation between the topology, as well as a novel class of the distributed averaging consensus algorithm.

While they delivery the same performance, the development of the algorithm comes from different grounds and act differently on different topologies. The believe propagation performs well with less iteration when the network is sparse while the distributed consensus performs well when the network is dense. Another different is that the distributed consensus' final output is a symmetrical result while our BP algorithm requires exact n iteration for the exact output as the centralized one, and n is the diameter of the network, aka, a longest of the shortest distances between any two nodes. We give proofs showing that the it can be done within n iterations with the exact result.

Then we further proposed a quantized version called tri-state message passing where we use 3-state quantized values instead of soft information, to save energy. -1 refers to negative decision, 0 refer to unsure and 1 refer to positive decision in each node. We further reduce the energy use by staying silent when the message is 0. We also conduct the capacity-assist analysis to find an optimal decision threshold that minimize the probability of errors. The simulation show that the method perform better than the case of no cooperation.

In the final part, which is the localization problem. We model the problem as unit disks graph (with integer alignment), formulate as an optimization problem and try to solve for the unique solution of the network. We proposed a search algorithm that make use of a side information about non-connectivity as a part that helps to eliminate infeasible solutions and exclude that from the search tree. This reduce the

search space significantly.

Also without the non-connectivity constraints, the unique solution can be related to the properties of the graph in the followings way. 3-connected graph is a necessary condition and 6-connected graph is a sufficient condition. If we include then non-connectivity constraint, these properties do not hold any more. It is, from the best of my knowledge, unknown and still an open problem to find out what is the necessary and sufficient condition for the network to have a unique solution. The simulation show significant complexity reduction when the non-connectivity constraint is imposed.

Appendices

Appendix A

Proof Of The Convergence of BP algorithm

A.0.1 Matrices and Graphs representation of the algorithm

The message passing algorithm from a node point of view.

$$m_{ij}^{(t)} = \phi_i + \sum_{k \in N(i) \backslash j} m_{ki}^{(t-1)} \tag{A.1}$$

$$b_i = \phi_i + \sum_{k \in N(i)} m_{ki} \tag{A.2}$$

Graph Representation

The BP algorithm is originated/formulated from any undirected graph. Here we represent the whole iterative process in a graphical form. We started with the original

undirected graph, then transform this into directed graph and associate the variable b_i with node i and messages m_{ij} with the edges. Below is the summary of a method to transform the original graph to our first directed graph representation. We call our original directed graph G, so in this case we call our directed representation as G'.

- Begin with any undirected graph, here we use fig. 3.4
- Replace edges in the original undirected with 2 directed edge labelled as m_{ij}. This two edges correspond to the messages passed from one node to another. Figure 3.5 is the result as G'.

Notice that in eqn. A.1, the new messages also exclude the incoming message in the same direction the outgoing message is going to be passed to. To capture this behavior, we transform edges in G' and represent m_{ij} as node instead. Here is the procedure

- Starting from G'
- Construct a new graph $\hat{G}$ where nodes are labelled as m_{ij} which is the one-on-one mapping from the message.
- Connect a direct link from m_{ki} to m_{ij}, $k \neq j$, aka, connect any incoming messages from surrounding node except the one that the message is pointing to in the original graph. If we consider nodes labeled as m_{ij} only, it can be depicted in fig. 3.6.

Now we can categorize the graph into to groups of nodes, one is the original nodes associated with b_i, we call them belief nodes. Then the new nodes we can call them

message nodes. Notice that at b_i has both outgoing and incoming flow, the out going flow in this case is actually the initial value ϕ_i, while the incoming flow is the message for evaluating b_i.

We further split b_i nodes group into ϕ_i nodes and b_i nodes to associate only one variable in one node. So the complete constructed graph suitable for our analysis is depicted as in fig. 3.7 and 3.8. We call this graph $\bar{G}$. This graph has 3 groups of nodes, m_{ij} nodes, ϕ_i nodes and b_i nodes, denoted as $\bar{G}_m$,$\bar{G}_\phi$ and $\bar{G}_b$ respectively. Later on we show that these 3 groups of graph can be represented in matrices forms.

Lemma A.0.1. *If G is a tree graph, $\bar{G}_m$ is acyclic directed graph.*

Proof. We prove this by an induction similar to proving a tree graph has no cycle. By starting from tree of size 2 and assume tree of size n has no cycle. Then finally show that a tree of size n+1, which is constructed by adding a new node and connect a edge between this one to the existing graph, also has no cycle.

1. Starting from the original G as a tree of two nodes and then transform this to $\bar{G}_m$. This can be easily verified as in fig. 3.7.

2. Now we assume that the original $\bar{G}$ of size n is also transformed into an acyclic graph $\bar{G}_m$ of size $2|n-1|$.

3. Then construct the G graph by connecting a new node j to node an existing node i in the graph. This is equivalent to adding new two nodes of m_{ij} and m_{ji} in $\bar{G}$.

4. To complete the process, we connect all m_{ki} from the existing $\bar{G}_m$, with the directed edges, to node m_{ji}. Since the existing graph has no cycle adding new

node with all incoming edges won't create a cycle. Similarly, connecting the new m_{ji} to all m_{ik}, won't create a cycle.

5. So now we have created $\bar{G}_m$ with size $2|n-1|$ without any cycle. This concludes that the $\bar{G}_m$ in general is acyclic directed graph.

□

Lemma A.0.1. *If G is a tree graph with diameter D, $\bar{G}_m$ has $D' = D - 1$ maximum walks.*

Proof. Similar to the induction stated above we can show that if the tree is size two, this lemma holds. Then we assume that the tree is size n and the lemma is true. Suppose the maximum path ends at m_{ij} we construct an $n+1$ tree by adding a node k at j.

This is equivalent to adding m_{jk} to the path ending at m_{ij}, hence increase the maximum walks by 1. Since m_{kj} does not connected to m_{jk}, the maximum walks for $n+1$ tree is $D'+1$. This concludes the proof. □

Lemma A.0.1. *If G is a tree graph, $\bar{G}$ is acyclic directed graph.*

Proof. We already show that $\bar{G}_m$ is acyclic and $\bar{G} = \bar{G}_\phi + \bar{G}_m + \bar{G}_b$. Connecting $\bar{G}_\phi$ to $\bar{G}_m$ won't create any cycle because all edges of $\bar{G}_\phi$ are out going direction to $\bar{G}_m$.

Similarly, we can merge $\bar{G}_b$ into the main graph without creating any cycle because $\bar{G}_b$ consist of all incoming edges from $\bar{G}_m$. Thus, the $\bar{G}$ is acyclic. □

Lemma A.0.1. *If G is a tree graph, there is at least 1 and only 1 path for any pair of nodes between $\bar{G}_\phi$ to $\bar{G}_b$*

Proof. We prove this by two facts, we first show that there exist a path between any node from $\bar{G}_\phi$ to $\bar{G}_b$ and then show that this is the only path.

1. For G of size n, there will be n nodes of $\bar{G}_\phi$ and n nodes of $\bar{G}_b$. We show that we can pick any node ϕ_i in $\bar{G}_\phi$ and any b_j in $\bar{G}_b$, we started from the original tree G, it is obvious from the property of a connected tree that there is a walk and only walk from node i to j.

   Suppose the path is $i \to k... \to l \to j$. Follow the same procedure when we construct $\bar{G}$ from G, then the corresponds path in $\bar{G}$ is $\phi_i \to m_{ik} \to m_{k...} \to ... \to m_{lj} \to b_j$.

2. Suppose there is another path $\bar{G}$ is $\phi_i \to m_{ik} \to m_{k...} \to ... \to m_{lj} \to b_j$ where at least $m_{k...}$ is not in the previous path. Then node k in the original tree G is not in the original undirected path. If there was another path in $\bar{G}$ through k then there must be the corresponding path in G through node k. This contradicts with the property of a tree.

□

Matrices representation

It is well known that any graph can be represented in a matrix, by adjacency matrix, node-edge-incident matrix, etc. BP algorithm, which is constructed based on graph, also has matrices representation. We will show that these matrices are actually representing exactly the same thing as its graphical representation. First we define some useful graph's related matrices and some theorems.

Definition A.0.1. *Adjacency Matrix: Let W be the $n \times n$ adjacency matrix of a graph G. Element $w_{ij} = 1$ if and only if the an edge $(i, j) \in G$. All other elements are zero. If G is directed graph, a row of W lists the nodes at the tip of the outgoing edges while a column of W lists the nodes at the tail of the incoming edges.*

Lemma A.0.1. *For Arbitrary nonnegative power (W^t) , an element w_{ij} is the number of paths of length exactly t from i to j [46].*

Proof. Consider the power W^2 of an adjacency matrix. Element w_{ij}^2 is the dot product $W(i,:)W(:,j)$. The k^{th} term in dot product equals 1 if and only if $W(i,k) = W(k,j) = 1$. In other words, it is 1 if and only if there is a path from $i \to k \to j$. The dot product thus counts the number of paths of length exactly 2 from i to j. Since $W^3 = W^2W$ and any arbitrary $W^{n+1} = W^nW$ can be proven by induction using the same argument similar to the case of W^2. □

Corollary A.0.1. *$W^t \neq 0$ for all t if and only if G has at least a cycle. Inversely, if G has no cycle, then $W^t = 0$ for all $t >$ the longest path in the graph.*

Definition A.0.1. *Edge-Node Incident Matrix: Let A be the $E \times n$ incident matrix of a directed graph G. Element $a_{ij} = 1$ if and only if the edge i is an incoming edge to node j, $a_{ij} = -1$ if it is an outgoing edge. All other elements are zero. We also define the $|A^-|$ as outgoing part of A, e.g. an element $a_{ij} = 1$ if and only if the edge i is an outgoing edge from node j, otherwise 0. Similarly, we define $|A^+|$ as the incoming part only.*

Consider eqn. (A.1), it can be written in a matrix form as follows.

Let M be a column vector of m_{ij} for all $i \neq j$; M is a matrix of size $1 \times l$, $l = 2|E|$. M is corresponding to the node of $\bar{G}_m$. Then we have

$$M^{(t)} = \Phi_l + WM^{(t-1)} \tag{A.3}$$

where W is a $l \times l$ directed-adjacent matrix whose elements $w_{kl} = 1$ when m_l^{t-1} contributes to the k^{th} message m_k^t. W represents the adjacency matrix of $\bar{G}_m$.

Φ_l is a column vector of size $1 \times l$ of the initial values. In fact,

$$\Phi_l = |A^-|\Phi_n$$

where $|A^-|$ edge-node incident of $\hat{G}$ or equivalently the adjacency matrix of nodes from $\bar{G}_\phi$ to $\bar{G}_m$. $|A^-|$ is of size $l \times n$. Φ_n is an $1 \times n$ column vector representing the initial values.

First step, M^0=Φ_l then

$$\begin{aligned}
M^{(1)} &= \Phi_l + W\Phi \\
M^{(2)} &= \Phi_l + WM^{(1)} \\
&= \Phi_l + W(\Phi_l + W\Phi_l) \\
&= \Phi_l + W\Phi_l + W^2\Phi_l \\
M^{(t)} &= (I + W + W^2 + W^3 + ... + W^t)\Phi_l
\end{aligned} \tag{A.4}$$

In the end, the messages will be combined to make the final decision. It also takes

the following form

$$B = \Phi_n + \left(|A^+|\right)^T M \tag{A.5}$$

$$B = \Phi_n + \left(|A^+|\right)^T (I + W + W^2 + W^3 + ... + W^t)|A^-|\Phi_n \tag{A.6}$$

Where B is a column vector of size $1 \times n$ storing the belief values for the nodes in $\bar{G}_b$. $|A^+|$ is the adjacency matrix between $\bar{G}_m$ to $\bar{G}_m$.

It is clear that eqn. (A.6) is equivalent to the graph representation $\bar{G}$.

Convergence of the algorithm

Lemma A.0.1. *Geometric Series of Matrices [25]: The geometric series of matrices* $\sum_{t=0}^{\infty} W^t$ *converges if and only if.* $|\lambda_i| < 1$ *for each eigenvalue* $|\lambda_i|$ *of* W*. Then we have*

- $|\lambda_i| < 1$ *implies* $(I - W)^{-1}$ *exists.*
- $\sum_{t=0}^{\infty} W^t = (I - W)^{-1}$.
- $\sum_{t=0}^{n} W^t = (I - W)^{-1}(I - W^n)$.

In general, the theorem holds when $|\lambda_i| = 0$.

Lemma A.0.1. *A graph* G *is an acyclic directed graph if and only if its adjacency matrix has all eigenvalues equal to zeros [26].*

Theorem A.0.1. *If a graph is tree-like, then the messages* M *in BP algorithm converges at exactly* D' *iterations.*

Proof. Consider an undirected tree graph G, it can be converted to $\bar{G}$.

1. We already show previously that $\bar{G}$ is acyclic directed graph. Hence W has all eigenvalues equal to zero.

2. Hence the series $W + W^2 + ... + W^t$ is finite and M converges.

3. From lemma A.0.1 and corollary A.0.1, one can show that $W^t = 0$ when $t > D'$.

4. Thus $\sum_{t=0}^{\infty} W^t$ converge exactly at $t = D'$ iterations.

□

The W^t has an interpretation; an entry w_{ij} gives the number of walk of length t from i to j. Thus W^t, in this case, indicates how the local information Φ_l propagates from one m_l to another by using exactly t hops. It can be observed that in order for a local information to propagate through out the network, $t = d$ iteration is required, where d is the diameter of the connected graph.

Next, we have to show that at the iteration $t \geq D'$, each node's value will be equal to the sum of the initial values of all nodes. That is any element b_i in B will be equal to the sum of all element in Φ_n, e.g. the sum or all log-likelihood ratio. And this value is, in fact, equal to the optimal centralized algorithm.

Theorem A.0.1. *Any element b_i in B will be equal to the sum of all element in Φ_o, e.g.*

$$\begin{aligned} B &= \Phi_n + \left(A^+\right)^T M \\ &= \left((\Phi_n)^T \left[111...1\right]_n\right) \left(\left[111...1\right]_n\right)^T \end{aligned} \tag{A.7}$$

Proof. 1. From lemma A.0.1, for a tree graph, there is at least 1 and only 1 path for any pair of nodes between $\bar{G}_\phi$ to $\bar{G}_b$.

2. From the property of the node-to-node tree in figure 3.5, it implies that the value b_i will be connected to the value Φ_{0i} for any i at most 1 time. And it also implies that b_i is connected to all Φ_{0i}.

3. From $W + W^2 + ... + W^t$ mean that the propagated value Φ_0 from any node will either reach a node b_i in 1 iteration, 2 iterations .. t iterations respectively. It implies that Φ_0 must reach b_i at least 1 time.

4. Each Φ_0 reaching to b_i at least 1 and at most 1 time means it reaches b_i exactly 1 time.

5. Hence, the final value of each element in B is in fact the sum of the initial value from each node. And this concludes the claim.

□

Theorem A.0.1. *If BP runs on undirected tree graph, after D' iterations, the belief values on all nodes are equal to $\sum_1^n \phi_i(x)$.*

Proof. Theorem A.0.1 and A.0.1 combined proves this theorem. □

Bibliography

[1] E. C. van der Meulen, Three-terminal communication channels, Adv. Appl. Prob., vol. 3, pp. 120-154, 1971.

[2] T. M. Cover and A. A. El Gamal, Capacity theorems for the relay channel, IEEE Trans. Inform. Theory, vol. IT-25, pp. 572-584, Sept. 1979

[3] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, Network information flow, IEEE Trans. on Information Theory, vol. 46, pp. 1204-1216, 2000.

[4] T. Migler, K. E. Morrison, and M. Ogle, "Weight and Rank of Matrices over Finite Fields," Mar. 2004.

[5] A. T. Campo, and A. Grant, "On Random Network Coding for Multicast," Feb. 2007.

[6] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "Toward a Random Operation of Networks," pp.1-30.

[7] J. Blomer, R. Karp, and E. Welzl, "The Rank of Sparse Random Matrices Over Finite Fields," Random Structures and Algorithms 10 (1997), pp.407-419.

[8] S. Acedanski, S. Deb, M. Medard, and R. Koetter, "How Good is Random Linear Coding Based Distributed Networked Storage?"

[9] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting," pp.1-6.

[10] M. Luby, "LT codes," *Proc. Foundations of Computer Science*, 2002.

[11] X. Bao, and J. Li, "Matching code-on-graph with network-on-graph: Adaptive network coding for wireless relay networks," *Proc. Allerton Conf. on Commun.,*

[12] P. K. Varshney, Distributed Detection and Data Fusion. New York: Springer-Verlag, 1996.

[13] C. Sun, W. Zhang and K. Ben, "Cluster-based cooperative spectrum sensing in cognitive radio systems," *IEEE Int. Conf. Commun.*, pp 24-28, June 2007.

[14] M. Gandetto, and C. Regazzoni, "Spectrum sensing: A distributed approach for cognitive terminals," *IEEE J. Sele. Area in Commun.*, pp. 546-557, April 2007.

[15] J. Ma and Y. Li, "Soft combination and detection for cooperative spectrum sensing in cognitive radio networks," *IEEE Globecom*, pp. 3139-3143, Nov. 2007.

[16] D. Cabric, A. Tkachenko, and R. W. Brodersen, "Experimental study of spectrum sensing based on energy detection and network cooperation," *ACM Intl. Conf Proc Series*, Vol. 222, No. 12, 2006.

[17] S. He, and C. Xiang, "A distributed cooperation spectrum sensing scheme in cognitive radio," *Intl. Conf. Wireless Commun., Net. and Mobile Computing*, 2008. 1-5.

[18] S. Zarrin, and T. J. Lim, "Belief propagation on factor graphs for cooperative spectrum sensing in cognitive radio," *New Frontiers in Dynamic Spectrum Access Networks*, 2008. 1-9.

[19] H. V. Poor, *An Introduction to Signal Detection and Estimation*, 2nd ed. Springer-Verlag, New York, 1994.

[20] S. M. Kay, "Fundamentals of Statistical Signal Processing: Detection Theory", vol. 2. Prentice Hall, 1998.

[21] L. Xiao, S. Boyd, and S.-J. Kim,"Distributed average consensus with least-mean-square deviation",Journal of Parallel and Distributed Computing, 67(1):33-46, 2007.

[22] Z. Li, F. Richard Yu and M. Huang, "Distributed Spectrum Sensing in Cognitive Radio Networks",WCNC 2009 proceedings.

[23] M. Yildizy, T. Aysaly and K. Barner, "In-Network Cooperative Spectrum Sensing", 17th European Signal Processing Conference (EUSIPCO 2009).

[24] G. Xiong and S. Kishore, Analysis of distributed consensus time synchronization with Gaussian delay over wireless sensor networks, submitted to EURASIP Journal on Wireless Communications and Networking.

[25] Charles Cullen,"Matrices and Linear Transformations", Addison-Wesley, Reading, Mass., 1966.

[26] D. M. Cvetkovic, M. Doob and H. Sachs, "Spectra of Graphs", third ed., Barth, Heidel-berg, 1995.

[27] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference", "Morgan Kaufmann", San Mateo, CA, 1988.

[28] Yedidia, Jonathan S. and Freeman, William T. and Weiss, Yair,"Understanding belief propagation and its generalizations", "Exploring artificial intelligence in the new millennium", pp. 239-269, 2003.

[29] Shridhar Mubaraq Mishra and Anant Sahai and Robert W. Brodersen,"Cooperative Sensing among Cognitive Radios", In Proc. of the IEEE International Conference on Communications (ICC), pp. 1658-1663, 2006.

[30] Xia, Wenfang and Wang, Shu and Liu, Wei and Cheng, Wenqing, "Cluster-based energy efficient cooperative spectrum sensing in cognitive radios", WiCOM'09: Proceedings of the 5th International Conference on Wireless communications, networking and mobile computing, pp. 1641-1644, 2009.

[31] Chunhua Sun and Wei Zhang and Ben, K., "Cluster-Based Cooperative Spectrum Sensing in Cognitive Radio Systems", Communications, 2007. ICC '07. IEEE International Conference on, pp. 2511-2515, 2007.

[32] Zarrin, S. and Teng Joon Lim, "Belief Propagation on Factor Graphs for Cooperative Spectrum Sensing in Cognitive Radio", New Frontiers in Dynamic Spectrum Access Networks, 2008. DySPAN 2008. 3rd IEEE Symposium on, pp. 1-9, 2008.

[33] Jun Ma and Guodong Zhao and Ye Li, "Soft Combination and Detection for Cooperative Spectrum Sensing in Cognitive Radio Networks", Wireless Communications, IEEE Transactions on, pp. 4502-4507, 2008.

[34] Wenzhong Wang and Luyong Zhang and Weixia Zou and Zheng Zhou, "On the distributed cooperative spectrum sensing for cognitive radio", Communications and Information Technologies, 2007. ISCIT '07. International Symposium on, pp. 1496-1501, 2007.

[35] Zhiqiang Li and Yu, F.R. and Minyi Huang, , "A Distributed Consensus-Based Cooperative Spectrum-Sensing Scheme in Cognitive Radios", Vehicular Technology, IEEE Transactions on, v. 59, no. 1, pp. 383-393, jan, 2010.

[36] P. Kaewprapha, R. Wu, B. C. Ng, T. J. Li, "Cooperative Spectrum Sensing for Cognitive Radios: Bounds and Algorithms", WCNC'10: Wireless communications and networking conference,2010.

[37] G. Mao, B. Fidan, B. Anderson, "Wireless sensor network localization techniques", *Computer Networks,* vol. 51, no. 10, 2007, pp 2529-2553.

[38] J.E. Graver, B. Servatius and H. Servatius, "Combinatorial rigidity", *American Mathematical Society,* 1993.

[39] J. Aspnes, T. Eren, D.K. Goldenberg, A.S. Morse, W. Whiteley, Y.R. Yang, B.D.O. Anderson, P.N. Belhumeur, "A Theory of Network Localization," *IEEE Trans Mobile Computing*, vol.5, no.12, 2006, pp.1663-1678.

[40] H. Breu and D.G. Kirkpatrick, "Unit Disk Graph Recognition is NP-Hard, *Computational Geometry. Theory and Appli.*, vol. 9, 1993.

[41] J. Aspnes, D. Goldenberg, Y.R. Yang, "On the Computational Complexity of Sensor Network Localization", *Algorithmic Aspects of Wireless Sensor Networks.* Springer, 2004. pp 32-44.

[42] A.A. Kannan, M Guoqiang, B. Vucetic, "Simulated Annealing based Wireless Sensor Network Localization with Flip Ambiguity Mitigation," *IEEE Vehicular Tech. Conf.* 2006, pp.1022-1026.

[43] Biswas, P. and Ye, Y. "Semidefinite programming for ad hoc wireless sensor network localization", *3rd Intl. Symp. Inf. Processing in Sensor Net.* 2004, pp 46-54.

[44] N. Bulusu, J. Heidemann, D. Estrin, "GPS-less low-cost outdoor localization for very small devices," *IEEE Personal Commun.* vol.7, no.5, pp.28-34, Oct 2000

[45] D. Niculescu, B. Nath, "Ad hoc positioning system (APS)," *IEEE GLOBECOM*, 2001.

[46] http://www8.cs.umu.se/kurser/5DA002/HT09/lectures/C4.pdf

Vita

Phisan Kaewprapha received Bachelor Degree in Computer Engineering from Chiang Mai University, Chiang Mai, Thailand in 2001. After graduation he worked at company in Bangkok, Thailand started as a programmer. His latest position there was assistant project leader.

Phisan was awarded the Royal Thai Government scholarship to pursue higher education. He then join the Department of Electrical and Computer Engineering, Lehigh University in 2005. He was working with Prof. Tiffany Jing Li Research Group. He finished the master degree in 2007 and continued in Ph.D program in Electrical and Computer Engineering.

Phisan research interests are centered around the area of distributed and parallel computing network, including communication designs, algorithm designs for network computing. Some specific area of the applications have been in his interests. Those areas are but not limit to cooperative data communication, distributed spectrum sensing, ad-hoc network localization, distributed data storage, etc.

After graduation, Phisan will work at Thammasat University, Bangkok, Thailand as a lecturer, where he will commit to contribute to the research community as well as teaching.