Optimal Monitoring in Multi-Channel Multi-Radio Wireless Mesh Networks

Dong-Hoon Shin and Saurabh Bagchi
Dependable Computing Systems Lab
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907, USA
{shin39, sbagchi}@purdue.edu

ABSTRACT

Wireless mesh networks (WMN) are finding increasing usage in city-wide deployments for providing network connectivity. Mesh routers in WMNs typically use multiple wireless channels to enhance the spatial-reuse of frequency bands, often with multiple radios per node. Due to the cooperative nature of WMNs, they are susceptible to many attacks that *cannot* be defeated by using traditional cryptographic mechanisms of authentication or encryption *alone*. A solution approach commonly used for defending against such attacks is *behavior-based detection* in which some nodes overhear communication in their neighborhood to determine if the behavior by a neighbor is legitimate. It has been proposed to use specialized monitoring nodes deployed strategically throughout the network for performing such detection. The problem that arises is where to deploy these monitoring nodes, how to minimize their number, and which channels to tune their radios to, such that the maximum part of the network can be covered. This problem has been solved for single channel networks by a greedy approximation algorithm since the exact solution is NP-hard. The greedy algorithm achieves the best performance, in terms of the worst case, possible among all polynomial-time algorithms provided that $P \neq NP$. In this paper, we solve the problem for multi-channel multi-radio WMNs. The intuitive extension of the greedy algorithm destroys the property of best performance. Instead, we formulate the problem as an integer linear program, solve its linear program relaxation, and then use two *rounding* techniques that we develop by adapting existing rounding schemes. We thereby present two approximation algorithms. The first, computationally-light algorithm, called *probabilistic rounding algorithm* gives an *expected* best performance in the worst case. The second, called *deterministic rounding algorithm* achieves the best worst-case performance in a *deterministic* manner. To evaluate how the three algorithms perform in practice, we simulate them in random networks and scale-free networks.

MobiHoc'09, May 18–21, 2009, New Orleans, Louisiana, USA.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Complexity of proof procedures*

General Terms

Algorithms, Security, Theory

Keywords

Wireless Mesh Networks, Multi-Channel Multi-Radio Wireless Networks, Security Monitoring, Approximation Algorithm, LP Rounding

1. INTRODUCTION

Wireless mesh networks (WMN) are finding increasing usage in city-wide deployments for providing network connectivity. In these networks, mobile devices connect to the internet gateways through mesh routers, which are typically stationary devices. Mesh routers forward packets *en route* to the gateways and typically use multiple wireless channels to enhance the spatial-reuse of frequency bands, often with multiple radios per node.

WMNs are vulnerable to a wide range of security attacks that are more severe and easier to launch in these networks than in their wired counterparts. An adversary can physically capture mesh routers and tamper with them [10] since they are often deployed in insecure locations. The compromised mesh routers can be exploited to launch a variety of attacks. Fundamentally, WMNs rely on cooperation of other nodes and this makes them vulnerable to malicious routers. A well-studied example is the malicious behavior at MAC layer where a malicious node does not follow the back-off rule for accessing the channel [8].

Many of these attacks cannot be defeated by using traditional cryptographic mechanisms of authentication or encryption *alone*. A simple example of this is that compromised nodes fabricate garbage packets that reach the end point and are then dropped, since they fail the authentication test; however, such garbage packets drain network resources since intermediate nodes on the route spend their resources forwarding the packets. An approach used to detect this class of attacks, which does not rely on cryptographic techniques, is *behavior-based detection*. In this, nodes observe the communication behavior of other nodes. In gen-

eral, the behavior being monitored can be other than communication behavior, e.g., sensing behavior to see if sensed data is accurate. Leveraging the one-hop broadcast nature of wireless communication, the monitoring node, which is a one-hop neighbor of the node being *verified*, overhears the communication of its neighbors and judges the behavior. For the example above, monitoring nodes verify that the back-off time of nodes follows a legitimate pattern. As another example, in case of malicious traffic injected into the network, such as worm traffic in a sensor network [13], the monitoring node can instruct intermediate nodes to drop such traffic, thereby saving resources.

A mechanism proposed in the literature is to have *specialized monitoring nodes* deployed throughout the network for validating the behavior of *normal nodes* in the network [12]. This takes the place of the more intuitively appealing architecture of every node participating in monitoring. The latter design is susceptible to framing of legitimate nodes due to erroneous reports by malicious nodes. The quorum-based solution [7] only works well under relatively high network densities, which are unlikely in most WMN deployments.

With specialized monitoring nodes, it would be desirable to optimize such nodes since they have specialized capabilities. The problem statement then is how to strategically place a given number of monitoring nodes in the network such that as large a fraction of the normal nodes as possible, ideally the entire set, is covered. For WMNs, the normal nodes can communicate on multiple channels and the monitoring nodes can also verify on multiple channels. Therefore, the problem statement is also to choose the channels that the monitoring nodes need to verify on, given the communicating channels of the normal nodes. The optimum placement and choice of channels is a function of the placement and traffic patterns of the normal nodes. When either changes, the solution will need to be re-evaluated. Alternatively, we consider a number of monitoring nodes deployed in the field and the problem statement is to decide which monitoring nodes to activate for achieving maximum coverage of the normal nodes since network intrusion detection is computationally expensive and energy-intensive. Note that the former problem statement can be mapped to the latter. Consider that the network is arranged as a grid with a given number, say k, of monitoring nodes available for placement on any grid point and we have to choose the grid points on which to place the monitoring nodes. Then, we can ask the following equivalent question: assuming that all the grid points have a monitoring node, how do we choose k out of them to maximize the coverage.

Previously, the problem has been solved for single channel networks [12], using a greedy strategy [6]. The determination of the optimal set of monitoring nodes is NP-hard in both single-channel and multi-channel networks since the maximum coverage problem, a well-known NP-hard problem, can be polynomially reduced to it. The greedy strategy of [12] has been shown to have an approximation ratio of $1 - 1/e$ [6]. Here, the approximation ratio, loosely speaking, is the minimum of ratios of the number of normal nodes covered by the given solution over the number of normal nodes covered by the optimal solution, the minimum being computed over all possible deployments. It has been shown that the greedy solution achieves the best approximation ratio possible among all polynomial-time algorithms unless $P = NP$ [4].

However, the intuitive extension of the single-channel solution to the multi-channel case leads to a greedy algorithm with an inferior approximation ratio of $\frac{1}{2}$. Therefore, we develop a different algorithm for the multi-channel case and show that our algorithm achieves the best approximation ratio $(1 - 1/e)$ possible among all polynomial-time algorithms unless $P = NP$. Our algorithm development takes the following path: we first formulate the problem as an integer linear program; we then relax the integer constraints and solve the resulting linear program (LP) relaxation, which takes polynomial time; Lastly, we adopt two LP rounding techniques—randomized rounding [11] and pipage rounding [1]—to achieve feasible solutions to our problem, which obey the integer constraints. We develop a *Probabilistic Rounding Algorithm* (PRA) which is computationally lightweight and guarantees the *expected* approximation ratio of $1 - 1/e$. Next, we develop a *Deterministic Rounding Algorithm* (DRA) which guarantees the approximation ratio in a *deterministic* manner to be $1 - 1/e$.

To evaluate how our solutions fare in practice, we perform simulations for different networks—random and scale-free—under different placements and densities. We compare the performance of the three algorithms—the greedy algorithm, PRA, and DRA—with the optimal value of LP relaxation, which we use an upper bound for the optimal value of our problem.

The rest of the paper is organized as follows. Background of LP rounding is introduced and related work is reviewed in Section 2. The problem formulation is described in Section 3. Hardness of the problem and greedy approximation algorithm are presented in Section 4. Two approximation algorithms using LP rounding are presented in Section 5. Complexity analysis is provided in Section 6. Performance evaluation through simulation is presented in Section 7. Finally, we conclude our work and discuss future work in Section 8.

2. BACKGROUND AND RELATED WORK

We first introduce a technique for designing approximation algorithms—LP rounding. We then review the previous works related to this paper.

2.1 LP Rounding

We often face optimization problems that can be formulated as integer linear programs (ILP). ILPs are in many practical situations NP-hard. Hence, we cannot solve such ILPs in polynomial time. Then, instead of exact solutions, we seek to find *approximate* ones achievable in polynomial time. There is a technique called *LP rounding*, which is a highly effective technique for designing approximation algorithms with proven performance guarantees [1]. The typical steps involved in finding the approximate solution are:

1) Formulate an optimization problem as an ILP
2) Transform the ILP to an LP relaxation by removing the integral constraints
3) Solve the LP relaxation (using one of many existing polynomial algorithms)
4) Round the optimal solution of LP relaxation, i.e., convert any non-integral values to integral values to obtain a *feasible* solution to ILP

In the fourth step, called *rounding*, there are two distinct approaches—*deterministic* and *randomized*. In this paper, we develop two rounding schemes derived from existing algorithms, `SAMPLING` [11] and `PIPAGE` [1], respectively. The former algorithm is randomized whereas the latter is deterministic, resulting in our two rounding schemes being randomized and deterministic, respectively.

2.2 Related Work

The selection of monitoring nodes for single-channel wireless networks, the most related work to ours, has been studied by Subhadrabandhu *et al.* in [12]. They solve the problem of optimal placement of monitoring nodes that execute intrusion detection modules, to maximize coverage in the network. They adapt a greedy algorithm in [6] for solving the problem, and it is the best approximation algorithm among all polynomial-time algorithms unless $P = NP$ [4].

The security issue in multi-channel multi-radio WMNs has been studied in [5, 9]. In [9], Naveed *et al.* identify vulnerabilities in channel assignment algorithms and briefly suggest possible solutions. Then, Haq *et al.* [5] propose a security mechanism to address the vulnerabilities exposed in [9].

Our mathematical formulation of the maximum coverage problem in multi-channel multi-radio WMNs (Section 3) was previously introduced in a generalized form, called *maximum coverage problem with group budget constraints* (our problem is its cardinality version) in [3]. In [3], Chekuri *et al.* present an approximation algorithm to solve the maximum coverage problem with group budget constraints, which is a simple extension from the greedy algorithm in [6]. They show that for the cardinality version, their algorithm solves the problem within a factor of $\frac{1}{2}$ of the optimum and their analysis is tight, which our performance result of GR-MCMC (Section 4) agrees with.

3. PROBLEM FORMULATION

We are given a set of n normal nodes $u_1, \ldots, u_n$. Node u_i has a_i radios called *normal radios*. We define $U = \{u_1^1, \ldots, u_1^{a_1}, \ldots, u_n^1, \ldots, u_n^{a_n}\}$, where u_i^j denotes the radio j of a normal node u_i. Each normal radio is tuned to a specific wireless channel, as determined by one of many existing channel assignment algorithms. The set U defines the set of normal radios to be verified by the monitoring nodes. There is a set of m monitoring nodes $v_1, \ldots, v_m$. Monitoring node v_i has t_i radios called *monitoring radios*. Each monitoring radio can be tuned to a channel $j = 1, \ldots, c$, where c is the number of wireless channels. We are also given a collection of subsets of U, $\mathcal{S} = \{S_{ij} : i = 1, \ldots, m,\ j = 1, \ldots, c\}$, where a *coverage-set* is defined as S_{ij}, the set of normal radios that can be covered by any radio of monitoring node v_i tuned on channel j. We say that a normal radio is *covered* by a monitoring radio if the latter can overhear the former's communication due to being tuned on the same channel. We will use the term "set" as a shorthand for "coverage-set" whenever we can do so without loss of clarity. We denote $\mathcal{S}_i = \{S_{ij} : j = 1, \ldots, c\}$, as a *group*. Our goal is to choose at most k sets from $\mathcal{S}$ with at most t_i ones from group $\mathcal{S}_i$ so as to maximize the number of normal radios covered by the selected sets. The constraint of k sets means that there are at most k number of monitoring radios that we can choose for verifying normal radios. This constraint is called the *total budget* constraint. The constraint of t_i is due to the observation that monitoring node v_i has t_i radios and our design that each radio will monitor a channel *continuously*, i.e., without hopping between different channels. This constraint is called the *group budget* constraint. If the k_i ($k_i \leq t_i$) sets, $S_{ij_1}, \ldots, S_{ij_{k_i}}$, in the group $\mathcal{S}_i$ are selected for a solution by any one of the algorithms presented by us here, then k_i radios of the monitoring node v_i will be tuned on the channels, $j_1, \ldots, j_{k_i}$. We refer to this problem as the *Maximum Coverage problem* with *Multiple Channels* (MCMC). For a cleaner exposition, we let all a_i's and t_j's be equal to 1. The general case is a simple extension of our solution detailed here. Therefore, we can simply let $u_i = u_i^1$ for notational convenience and denote each radio by its node since there is a one-to-one mapping between radios and nodes. We denote a special case of MCMC where all nodes (normal nodes and monitoring nodes) have a single channel and a single radio (MCMC with $c = 1,\ a_i = 1,\ \forall i = 1, \ldots, n,\ t_j = 1,\ \forall j = 1, \ldots, m$) as the *Maximum Coverage problem* with *Single Channel* (MCSC).

4. NP-HARDNESS OF PROBLEM AND GREEDY APPROXIMATION

Lemma 1. *MCMC is an NP-hard problem.*

PROOF. MCMC can be reduced to the *maximum coverage problem*[1] by setting $c = 1$. Thus, if the optimal solution to MCMC can be determined in polynomial time, then the maximum coverage problem can also be solved in polynomial time, which is a contradiction. ■

Then, it is reasonable to pursue an approximate solution achievable in polynomial time.

Definition 1. *A polynomial-time algorithm is said to be a* δ-approximation algorithm *for a maximization problem if for every instance of the problem, it delivers a solution that is at least* δ *times the optimum. Here,* δ *is referred to as the* approximation ratio.

Naturally, $\delta < 1$, and the closer it is to 1, the better. In this paper, we seek to find the answers to the following questions for our maximization problem, MCMC: What is the best approximation ratio attainable? How can it be achieved through a realizable algorithm?

We first consider MCSC, which is a special case of MCMC but still an NP-hard problem since it is exactly the maximum coverage problem. It is known that a simple greedy algorithm solves MCSC within a factor of $1 - (1 - 1/k)^k$ of the optimum [6], where k is the maximum number of monitoring nodes that can be selected. Let us denote it as GR-MCSC. GR-MCSC selects k sets in $\mathcal{S}$ by picking at each iteration the set that covers the maximum number of uncovered elements. It is proven in [4] that no polynomial-time algorithm can achieve higher approximation ratio than $1 - 1/e$ (≈ 0.632) provided that $P \neq NP$.

Lemma 2. *GR-MCSC is the* best *approximation algorithm for MCSC unless* $P = NP$.

PROOF. It follows from that $1-(1-1/k)^k > 1-1/e$ since $\lim_{k\to\infty}[1 - (1 - 1/k)^k] = 1 - 1/e$ and $1 - (1 - 1/k)^k$ is a decreasing function of k [6]. ■

[1]Given a set $U = \{1, \ldots, n\}$ and a collection of subsets of U, $\mathcal{C} = \{C_1, \ldots, C_m\}$, the *maximum coverage problem* is to select k of these subsets such that the cardinality of union of the selected subsets is maximized. It is an NP-hard problem. [6]

Algorithm 1 `GR-MCMC`

1: $I \leftarrow \{1, 2, \ldots, m\}$, $S'_{ij} \leftarrow S_{ij}$, $\forall i = 1, \ldots, m$, $\forall j = 1, \ldots, c$, $\mathcal{G} \leftarrow \emptyset$
2: **for** $l \leftarrow 1$ to k **do**
3: Find i_l, j_l such that $\left|S'_{i_l j_l}\right| = \max_{\forall i \in I,\, \forall j} \left|S'_{ij}\right|$
4: Do $G_l \leftarrow S_{i_l j_l}$, where G_l denotes the l-th set in $\mathcal{G}$
5: $I \leftarrow I \backslash \{i_l\}$
6: **for** each $i \in I$ **do**
7: **for** $j \leftarrow 1$ to c **do**
8: $S'_{ij} \leftarrow S'_{ij} \backslash S'_{i_l j_l}$
9: **end for**
10: **end for**
11: **end for**
12: **return** $\mathcal{G}$

We proceed to explore if the approximation ratio of $1-1/e$ is attainable for MCMC. Intuitively, we first generalize the greedy approach of GR-MCSC to MCMC and thereby propose our first algorithm, *GReedy algorithm for MCMC* (GR-MCMC). Note that a direct application of GR-MCSC by iterating it over all channels on all monitoring nodes can violate the group budget constraint. Basically, GR-MCMC operates similarly to GR-MCSC except that once a set S_{ij} is chosen from group $\mathcal{S}_i$, no other set from $\mathcal{S}_i$ is considered for further selection. We formally present GR-MCMC, denoted `GR-MCMC`, in Algo. 1. For each iteration of the `for` loop in line 2, we pick the set covering the maximum number of uncovered elements. We remove the already covered elements from the remaining sets in lines 6–10. We terminate the loop once k monitoring nodes are selected.

We are now interested in the performance of `GR-MCMC`. We first prove the following lemmas.

Lemma 3. *`GR-MCMC` is a $\frac{1}{2}$-approximation algorithm.*

PROOF. We define $\mathcal{H} = \{H_1, \ldots, H_k\}$ as the optimal selection and $\mathcal{H}_1$, $\mathcal{H}_2$ to be a partition of $\mathcal{H}$. Here, if $H_j \in \mathcal{H}$ has a same group index as that of some $G_i \in \mathcal{G}$, then $H_j \in \mathcal{H}_1$, and otherwise $H_j \in \mathcal{H}_2$. We similarly define $\mathcal{G}_1$, $\mathcal{G}_2$ to be a partition of $\mathcal{G}$. Consider a set $H_j \in \mathcal{H}_1$. There must exist some $G_i \in \mathcal{G}_1$ whose group index is same as that of H_j. `GR-MCMC` picks G_i at i-th iteration, which is the set of maximum number of uncovered elements among remaining sets at i-th iteration. The fact that G_i and H_j belong to same group implies that H_j is available at i-th iteration. Hence, we get that $|G_i - \cup_{l=1}^{i-1} G_l| \geq |H_j - \cup_{l=1}^{i-1} G_l| \geq |H_j - \cup_{l=1}^{k} G_l|$, and thereby it follows that $\sum_{i:G_i \in \mathcal{G}_1} |G_i - \cup_{l=1}^{i-1} G_l| \geq \sum_{j:H_j \in \mathcal{H}_1} |H_j - \cup_{l=1}^{k} G_l|$—$(*)$. Also, due to the property of `GR-MCMC` mentioned right above, it is true that $\forall G_i \in \mathcal{G}$, $\forall H_j \in \mathcal{H}_2$, $|G_i - \cup_{l=1}^{i-1} G_l| \geq |H_j - \cup_{l=1}^{i-1} G_l| \geq |H_j - \cup_{l=1}^{k} G_l|$, and therefore we get that $\sum_{i:G_i \in \mathcal{G}_2} |G_i - \cup_{l=1}^{i-1} G_l| \geq \sum_{j:H_j \in \mathcal{H}_2} |H_j - \cup_{l=1}^{k} G_l|$—$(**)$. Using $(*)$ and $(**)$, we can obtain that $\sum_{i=1}^{k} |G_i - \cup_{l=1}^{i-1} G_l| \geq \sum_{i=1}^{k} |H_i - \cup_{l=1}^{k} G_l|$—$(***)$. Since it is true that $|\cup_{i=1}^{k} G_i| = \sum_{i=1}^{k} |G_i - \cup_{l=1}^{i-1} G_l|$, and also that $\sum_{i=1}^{k} |H_i - \cup_{l=1}^{k} G_l| \geq |\cup_{i=1}^{k} (H_i - \cup_{l=1}^{k} G_l)| = |\cup_{i=1}^{k} H_i - \cup_{l=1}^{k} G_l)| \geq |\cup_{i=1}^{k} H_i| - |\cup_{l=1}^{k} G_l|$, it follows from $(***)$ that $|\cup_{i=1}^{k} G_i| \geq |\cup_{i=1}^{k} H_i| - |\cup_{i=1}^{k} G_i|$. Therefore, we finally get that $|\cup_{i=1}^{k} G_i| \geq \frac{1}{2} |\cup_{i=1}^{k} H_i|$. ■

Lemma 4. *The approximation ratio of $\frac{1}{2}$ of `GR-MCMC` cannot be improved further.*

PROOF. Consider an input instance for `GR-MCMC`: $S_{11} = \{u_1, \ldots, u_n\}$, $S_{12} = \{u_{n+1}, \ldots, u_{2n-1}\}$, $S_{21} = \{u_1, \ldots, u_{n-1}\}$, $S_{22} = \{u_{2n}\}$, and $k = 2$. Then, `GR-MCMC` outputs S_{11} and S_{22} while the optimal selection is S_{12} and S_{21}. Therefore, a ratio r of the solution value of `GR-MCMC` to the optimum is $r = \frac{n+1}{2(n-1)} = \frac{1}{2} + \frac{1}{n-1} > \frac{1}{2}$. We can make r arbitrarily close to $\frac{1}{2}$ by setting a large value of n. Thus, the approximation ratio of `GR-MCMC` cannot be greater than $\frac{1}{2}$. ■

Lemmas 3 and 4 lead to the following proposition.

Proposition 1. *`GR-MCMC` is a $\frac{1}{2}$-approximation algorithm and cannot be improved further.*

5. LP ROUNDING ALGORITHMS

From MCSC, we determine that the best possible approximation ratio is $1 - 1/e$ provided that $P \neq NP$. `GR-MCMC` achieves $\frac{1}{2}$. Thus, we have the unanswered question: does there exist an approximation algorithm that can achieve the ratio $1-1/e$? In this section, we present two approximation algorithms: PRA and DRA. PRA is a randomized algorithm and probabilistically attains the best approximation ratio $1 - 1/e$. On the other hand, DRA is a deterministic algorithm and deterministically achieves the best approximation ratio. Both algorithms employ the LP rounding technique, which was explained in Section 2.1.

5.1 ILP Formulation and LP Relaxation Solution

We first formulate MCMC as an ILP. We assign variables x_l, y_{ij} to $u_l \in U$ and $S_{ij} \in \mathcal{S}$, respectively. If S_{ij} is chosen for a solution, y_{ij} is set to 1; otherwise, it is set to 0. If $u_l \in U$ is covered by a solution, x_l is set to 1; otherwise, it is set to 0. For notational convenience, let us define index sets $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, c\}$, which will be used throughout this section. We now formulate MCMC as the following ILP:

$$\textbf{(ILP-MC)} \quad \max \quad \sum_{l=1}^{n} x_l \tag{1}$$

$$\text{subject to} \quad x_l \leq \sum_{i,j:\, u_l \in S_{ij}} y_{ij},\ \forall l \in \{1, \ldots, n\}, \tag{2}$$

$$\sum_{i=1}^{m} \sum_{j=1}^{c} y_{ij} \leq k, \tag{3}$$

$$\sum_{j=1}^{c} y_{ij} \leq 1, \quad \forall i \in I, \tag{4}$$

$$0 \leq y_{ij} \leq 1, \quad \forall i \in I,\ \forall j \in J, \tag{5}$$

$$0 \leq x_l \leq 1, \quad \forall l \in \{1, \ldots, n\}, \tag{6}$$

$$y_{ij} \in \{0, 1\}, \quad \forall i \in I,\ \forall j \in J. \tag{7}$$

The constraint (2) says that x_l is upper bounded by the number of sets chosen for a solution that have u_l in them. It seems that we need an additional constraint to restrict $x_l \in \{0, 1\}$. However, x_l will automatically be set to either 0 or 1 due to (6) and (7), and due to the fact that we would like to maximize x_l. The constraints (3) and (4) arise because we can choose at most k sets (total budget) with at most one in each group (group budget). Although (5) is unnecessary here due to (7), we keep it for the LP relaxation

since we need to still maintain (5) after relaxing the integral constraint (7). As expected due to the NP-hardness of MCMC (Lemma 1), ILP-MC cannot be solved in polynomial time.

Next, we perform LP relaxation by removing the integral constraint (7) from ILP-MC and denote it as LP-MC. Now y_{ij} can take any value in $[0, 1]$, including fractional values, and therefore loses its physical interpretation in ILP-MC. We can obtain the optimal solution to LP-MC through one of many existing polynomial LP solvers. We develop two rounding schemes, *probabilistic rounding scheme* and *deterministic rounding scheme*, and provide the algorithms, PRA and DRA, each using the corresponding rounding scheme as its subroutine. In fact, we should only round y_{ij}'s and not the x_l's. Rather, we define x_l naturally to be $\max\{y^*_{ij} : u_l \in S_{ij}\}$, where y^*_{ij} is the resulting value after rounding.

5.2 Probabilistic Rounding Algorithm

Probabilistic Rounding Scheme (PRS) treats the value of each variable as the probability of rounding the variable to 1. Let $\tilde{\vec{y}} = (\tilde{y}_{ij} : i \in I,\ j \in J)$ be the optimal solution to LP-MC and define Y_{ij} to be the resulting integral value of $\tilde{y}_{ij}$ after PRS runs. Since $\tilde{y}_{ij}$ is rounded in a probabilistic manner, Y_{ij} is a binary random variable. Then, Y_{ij}'s will satisfy the following properties:

(P1) $\Pr[Y_{ij} = 1] = \tilde{y}_{ij},\ \forall i \in I,\ \forall j \in J,$

(P2) $\sum_{i=1}^{m} \sum_{j=1}^{c} Y_{ij} \le k,$

(P3) $\sum_{j=1}^{c} Y_{ij} \le 1,\ \forall i \in I,$

(P4) $\Pr\big[\bigcap_{(i,j)\in H}\{Y_{ij} = 0\}\big] \le \prod_{(i,j)\in H} \Pr[Y_{ij} = 0],$
$\forall H \subseteq \big\{(i, j) : i \in I,\ j \in J\big\}.$

The properties (P1) and (P4) will be proved and used in the proof of Lemma 7 to show the performance of PRS. The other two properties (P2) and (P3) are necessary for the resulting solution after the execution of PRS to be feasible for ILP-MC. Their proofs will be given in the proof of Lemma 5.

To meet the four properties (P1)–(P4), PRS employs an existing algorithm, called `SAMPLING` [11]. Given a vector $(p_1, \ldots, p_t)$ such that $p_i \in [0, 1],\ \forall i = 1, \ldots, t$, and $\sum_{i=1}^{t} p_i$ is an integer, say p, `SAMPLING` generates a vector of 0's or 1's $(X_1, \ldots, X_t)$ satisfying the following properties:

(S1) $\Pr[X_i = 1] = p_i,\ \forall i \in \{1, \ldots, t\},$

(S2) $\Pr\Big[\big|\{i : X_i = 1\}\big| = p\Big] = 1,$ (*deterministically*, the number of X_i's being 1 is p),

(S3) $\Pr\left[\bigcap_{i\in S}\{X_i = 0\}\right] \le \prod_{i\in S} \Pr\left[X_i = 0\right],$
$\forall S \subseteq \{1, \ldots, t\}.$

We now explain how PRS is developed through the use of `SAMPLING`. PRS (refer to Algo. 2) consists of two phases. In the first phase (line 2), the exact k' $(= \lceil \sum_{i=1}^{m} \sum_{j=1}^{c} \tilde{y}_{ij} \rceil)$ groups are selected using `SAMPLING` with the vector $\vec{y} = (\tilde{y}_1, \ldots, \tilde{y}_m, k' - \sum_{i=1}^{m} \tilde{y}_i)$, where $\tilde{y}_i = \sum_{j=1}^{c} \tilde{y}_{ij}$ and the last element is padded to make the total sum integral, which is a requirement of `SAMPLING`. In all cases, except when additional monitoring nodes give no coverage improvement, $k' = k$, i.e., solution of LP-MC will use the maximum allowable number of monitoring nodes. The last element Z_{m+1} in the output of `SAMPLING` is thrown away, leaving m elements again in the vector. In the second phase (lines 3–9),

Algorithm 2 `Probabilistic Rounding Scheme` (`PRS`)

1: $k' \leftarrow \lceil \sum_{i=1}^{m} \sum_{j=1}^{c} \tilde{y}_{ij} \rceil,\ \tilde{y}_i \leftarrow \sum_{j=1}^{c} \tilde{y}_{ij},\ \forall i = 1, \ldots, m,$
$\vec{y} \leftarrow (\tilde{y}_1, \ldots, \tilde{y}_m, k' - \sum_{i=1}^{m} \tilde{y}_i)$
2: $\vec{Z} = (Z_1, \ldots, Z_m, Z_{m+1}) \leftarrow$ `SAMPLING`$(\vec{y})$
3: **for** $l \leftarrow 1$ to m **do**
4: **if** $Z_l = 1$ **then**
5: $(Y_{l1}, \ldots, Y_{lc}) \leftarrow$ `SAMPLING`$(\tilde{y}_{l1}/\tilde{y}_l, \ldots, \tilde{y}_{lc}/\tilde{y}_l)$
6: **else**
7: $(Y_{l1}, \ldots, Y_{lc}) \leftarrow (0, \ldots, 0)$
8: **end if**
9: **end for**
10: **return** $\vec{Y} = (Y_{11}, \ldots, Y_{1c}, \ldots, Y_{m1}, \ldots, Y_{mc})$

Algorithm 3 `Probabilistic Rounding Algorithm` (`PRA`)

1: Formulate ILP-MC from a given MCMC
2: Transform ILP-MC into LP-MC by removing the integral constraint
3: Obtain the optimal solution $\tilde{\vec{y}} = (\tilde{y}_{ij} : i \in I,\ j \in J)$ to LP-MC (using one of many existing LP solvers)
4: **if** $\tilde{\vec{y}}$ is an integral vector **then**
5: $\vec{Y} = \tilde{\vec{y}}$
6: **else**
7: $\vec{Y} =$ `PRS`$(\tilde{\vec{y}})$
8: **end if**
9: **return** $\vec{Y}$

`SAMPLING` with the input $(\tilde{y}_{l1}/\tilde{y}_l, \ldots, \tilde{y}_{lc}/\tilde{y}_l)$ picks only one set in each group l selected in the first phase. The formal description of PRA, denoted `PRA`, using `PRS` as its subroutine is provided in Algo. 3.

We now show the feasibility and performance of the solution of `PRA`. We first prove the feasibility of the solution.

Lemma 5. *`PRA` presents a feasible solution $\vec{Y}$ to ILP-MC.*

Proof. If the condition in line 4 of `PRA` is true, then it is obvious that $\vec{Y}$ is a feasible solution to ILP-MC. Now, we consider the case when the condition in line 4 is not true. In line 2 of `PRS`, the number of 1's of the first m elements in $\vec{Z}$ is k' or $k' - 1$. In line 5 of `PRS`, for every $l \in I$ such that $Z_l = 1$, only one element in the vector $(Y_{l1}, \ldots, Y_{lc})$ has a value of 1 due to the fact that $\sum_{j=1}^{c} \tilde{y}_{lj}/\tilde{y}_l = 1$ and (S2) while for every $l \in I$ such that $Z_l = 0$, all the elements in the vector $(Y_{l1}, \ldots, Y_{lc})$ are 0. This proves the property (P3). Since $k' = \lceil \sum_{i=1}^{m} \sum_{j=1}^{c} \tilde{y}_{ij} \rceil \le \lceil k \rceil = k$, at most k or $k - 1$ elements are 1 and all the others are 0, leading to the property (P2). Thus, $\vec{Y}$ is a feasible solution to ILP-MC. ■

Before we show the performance of `PRA`, we introduce a useful lemma from [1] for proving it. This lemma will also be used for the proof of Lemma 12.

Lemma 6. *For $0 \le y_{ij} \le 1,\ \forall i \in I,\ \forall j \in J,$*

$$1 - \prod_{i,j:\ u_l \in S_{ij}} (1 - y_{ij}) \ge \big(1 - (1 - 1/p)^p\big) \min\{1, \sum_{i,j:\ u_l \in S_{ij}} y_{ij}\},$$

where $p = \Big|\big\{(i, j) : u_l \in S_{ij}\big\}\Big|$.

For our MCMC, p means the maximum number of monitoring nodes that can cover a given normal node.

Lemma 7. *The expected solution quality of* PRA *is at least* $1-(1-1/m)^m$ *times the optimal value of ILP-MC, where* m *denotes the number of monitoring nodes.*

PROOF. It follows from (S1) that $\forall i \in I,\ \forall j \in J,\ \Pr[Y_{ij}=1] = \Pr[Y_{ij}=1|Z_i=1]\cdot\Pr[Z_i=1] = (\tilde{y}_{ij}/\tilde{y}_i)\cdot\tilde{y}_i = \tilde{y}_{ij}$, proving the property (P1). We omit the detailed proof of (P4) here due to the limitation of space. However, one can verify that $\texttt{PRS}(\tilde{\vec{y}})$ is a specific way of implementing $\texttt{SAMPLING}(\tilde{\vec{y}})$ and therefore (S3) holds for Y_{ij}'s by PRS output. This proves (P4) to be true. We define $\{\tilde{x}_l, \tilde{y}_{ij} : l = 1,\ldots,n,\ i=1,\ldots,m,\ j=1,\ldots,c\}$ and $\tilde{z}$ to be the optimal solution to LP-MC and its optimal value, respectively. Let $X_l = \max\{Y_{ij} : u_l \in S_{ij}\}$. Then, we have the following equation:

$$\begin{aligned}
\Pr[X_l=1] &= 1-\Pr\left[\bigcap_{i,j:u_l\in S_{ij}}\{Y_{ij}=0\}\right] \\
&\geq 1-\prod_{i,j:\ u_l\in S_{ij}}\Pr[Y_{ij}=0] \quad \big(\text{from (P4)}\big) \\
&= 1-\prod_{i,j:\ u_l\in S_{ij}}(1-\tilde{y}_{ij}) \quad \big(\text{from (P1)}\big) \\
&\geq \big(1-(1-1/m)^m\big)\min\{1, \sum_{i,j:\ u_l\in S_{ij}}\tilde{y}_{ij}\} \\
&\qquad \text{(from Lemma 6 and } m\geq p) \\
&= \big(1-(1-1/m)^m\big)\tilde{x}_l.
\end{aligned}$$

The last equality follows from that $\{\tilde{x}_l, \tilde{y}_{ij}\}$ satisfies the constraints (2), (6), and that we would like to maximize $\tilde{x}_l$. We thereby have that $E\big[\sum_{l=1}^n X_l\big] \geq \big(1-(1-1/m)^m\big)\sum_{l=1}^n \tilde{x}_l = \big(1-(1-1/m)^m\big)\cdot\tilde{z}$. Since the optimal value $\tilde{z}$ of LP-MC is an upper bound of that of ILP-MC, the proof is completed. ■

Now, we state the following proposition.

Proposition 2. *The expected approximation ratio of* PRA *is* $1-1/e$*, which is the best possible approximation ratio unless* $P=NP$.

PROOF. Due to Lemma 7, it suffices for proving the above proposition to recall that $1-(1-1/m)^m > 1-1/e$, which has been shown in the proof of Lemma 2. ■

The value of $1-1/e$ is reached asymptotically when m grows to infinity. Thus, practically with a finite number of monitoring nodes, the expected worst-case performance of PRA is better than $1-1/e$ times the quality of the optimal solution.

5.3 Deterministic Rounding Algorithm

We now approach the problem of finding an algorithm that can provide the best possible approximation ratio in a deterministic manner. We develop DRA to solve MCMC. DRA uses a *Deterministic Rounding Scheme* (DRS), which we develop and which uses a previous algorithm called PIPAGE [1].

We first describe the algorithm PIPAGE. Suppose that we are given a bipartite graph $G=(P,Q;E)$, a function $F(\vec{x})$ defined on points $\vec{x}=(x_e : e\in E)$ of the $|E|$-dimensional cube $[0,1]^{|E|}$ and computable in polynomial time, and a function $p: P\cup Q \to \mathbb{Z}_+$, where $\mathbb{Z}_+$ denotes the set of positive integers. Consider a binary program of the following form:

$$\begin{aligned}
(\mathbf{BP})\quad max \quad & F(\vec{x}) && (8)\\
\text{subject to} \quad & \sum_{e\in E(v)} x_e \leq p(v), \quad v\in P\cup Q, && (9)\\
& 0\leq x_e\leq 1, \quad e\in E, && (10)\\
& x_e\in\{0,1\}, \quad e\in E, && (11)
\end{aligned}$$

where $E(v)$ denotes the set of edges that are connected to a vertex v. Note that the function $F(\vec{x})$ does not need to be linear. We say that a solution only satisfying (9) and (10) is *fractional* since some of its components may be non-integral. The algorithm PIPAGE takes a BP and a fractional $\vec{x}$ as its input and outputs a new fractional solution $\vec{x}'$.

Now, we describe how PIPAGE proceeds. If $\vec{x}$ is integral, then PIPAGE terminates and outputs $\vec{x}'=\vec{x}$. Suppose now that $\vec{x}$ is non-integral. Construct the subgraph $H_{\vec{x}}$ of G with the same vertex set and the edge set $E_{\vec{x}}$ defined by the condition that $e\in E_{\vec{x}}$ if and only if x_e is non-integral. If $H_{\vec{x}}$ contains cycles, then we set R to be one such cycle. If $H_{\vec{x}}$ is a forest, then we set R to be one path of $H_{\vec{x}}$ whose endpoints have degree 1. Since $H_{\vec{x}}$ is bipartite, in both cases R can be uniquely represented as the union of two matchings[2]. Let M_1 and M_2 denote those matchings. Define a new solution $\vec{x}(\epsilon,R)$ by the following rule: if $e\in E\backslash R$, then $x_e(\epsilon,R)=x_e$, otherwise $x_e(\epsilon,R)=x_e+\epsilon$ when $e\in M_1$, and $x_e(\epsilon,R)=x_e-\epsilon$ when $e\in M_2$. Set $\epsilon_1=\min\big\{\min_{e\in M_1}x_e,\ \min_{e\in M_2}(1-x_e)\big\}$ and $\epsilon_2=\min\big\{\min_{e\in M_1}(1-x_e),\ \min_{e\in M_2}x_e\big\}$. Let $\vec{x}_1=\vec{x}(-\epsilon_1,R)$ and $\vec{x}_2=\vec{x}(\epsilon_2,R)$. Set $\vec{x}'=\vec{x}_1$ if $F(\vec{x}_1)>F(\vec{x}_2)$ and $\vec{x}'=\vec{x}_2$ otherwise. Note that by the construction of PIPAGE, if $\vec{x}$ is non-integral, the number of integral components in $\vec{x}'$ is at least one greater than that in $\vec{x}$. Hence, we can transform a non-integral solution to an integral one satisfying (9)–(11) after iterating PIPAGE at most $|E|$ times.

We move on to describe DRS, which makes use of PIPAGE. We define $F(\vec{y})=\sum_{l=1}^n\big(1-\prod_{i,j:\ u_l\in S_{ij}}(1-y_{ij})\big)$. Note that ILP-MC is equivalent to maximizing $F(\vec{y})$ under the constraints (3), (4), and (7). This maximization problem consists of only the variables y_{ij}'s rather than two sets of variables—y_{ij}'s and x_l's as in the original ILP-MC. Thus, this is of the form of BP. However, due to having two constraints, total budget and group budget, it is impossible to make the problem into the form of BP as in (8)–(11). Therefore, we design DRS to operate in two phases. In the first phase, we round non-integral variables satisfying the group budget constraint while in the second phase, we round the remaining non-integral variables after the first phase satisfying the total budget constraint. When PIPAGE rounds in the first phase, it may violate the total budget constraint since it is not included as a constraint in the binary program of the first phase (refer to BP1 described below). Therefore, an important goal is to keep satisfying the total budget constraint in the first phase, which we achieve through modifying PIPAGE.

We define $\tilde{\vec{y}}=(\tilde{y}_{ij} : i\in I,\ j\in J)$ to be the optimal solution to LP-MC. We first explain the first phase of DRS. Consider a bipartite graph $G_1=(P_1,Q_1;E_1)$, shown in Fig. 1(a), where $P_1=\{1,\ldots,m\}$, $Q_1=\{1,\ldots,mc\}$, and $E_1=\{e_i=(p_i,q_i) : p_i=\lceil i/c\rceil\in P_1,\ q_i=i\in Q_1,\ i=$

[2]In a graph, a *matching* is a set of edges without common vertices.

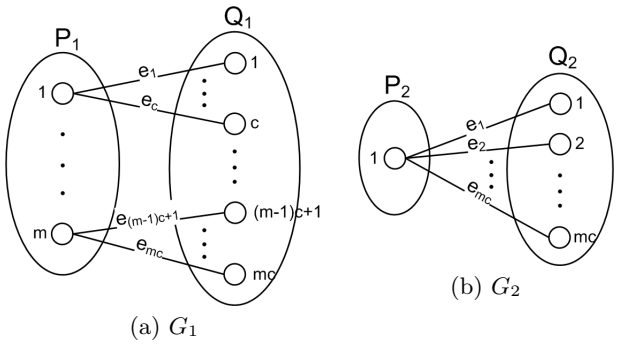


Figure 1: Bipartite graphs

$1, \ldots, mc\}$. We have a total of $|E_1| = mc$ edges in G_1 and assign variables to them such that y_{ij} is assigned to the edge $e_{(i-1)*c+j} = (i, (i-1)c + j)$. Build the binary program:

$$\textbf{(BP1)} \quad max \qquad F(\vec{y}) \tag{12}$$

$$\text{subject to} \qquad \sum_{j=1}^{c} y_{ij} \leq 1, \quad \forall i \in I, \tag{13}$$

$$0 \leq y_{ij} \leq 1, \quad \forall i \in I, \forall j \in J, \tag{14}$$

$$y_{ij} \in \{0, 1\}, \quad \forall i \in I, \forall j \in J. \tag{15}$$

In the first phase, DRS uses a new algorithm called `MOD-PIPAGE`, which is derived from `PIPAGE`. DRS calls `MOD-PIPAGE` with BP1 (which includes G_1) and $\tilde{\vec{y}}$ as inputs. Note that R in G_1 cannot be a cycle, thus it is constrained to be of length one or two. `MOD-PIPAGE` creates the sub-graph $H_{\tilde{\vec{y}}}$ from G_1 as before. But, different from `PIPAGE`, it considers a path R only if at least two edges are non-integral. This restricts R to be of length exactly two. `MOD-PIPAGE` exits when no such path R can be found. Hence, if there exists a path of length two in $H_{\tilde{\vec{y}}}$ satisfying the non-integral constraint from above, then `MOD-PIPAGE` will produce an output with at least one more integral component. The first phase of DRS iteratively runs `MOD-PIPAGE` and terminates when `MOD-PIPAGE` exits without making any change. We denote the resulting vector after the first phase as $\vec{y}^*$. Each vertex in P_1 of the graph G_1 now has at most one non-integral edge.

Next, we describe the second phase of DRS. Consider a bipartite graph $G_2 = (P_2, Q_2; E_2)$, shown in Fig. 1(b), where $P_2 = \{1\}$, $Q_2 = \{1, \ldots, mc\}$, and $E_2 = \{e_i = (1, i) : 1 \in P_2,\ i \in Q_2,\ i = 1, \ldots, mc\}$. We have in total $|E_2| = mc$ edges in G_2 and assign variables to them such that y_{ij} is assigned to the edge $e_{(i-1)*c+j} = (1, (i-1)c + j)$. Build the binary program:

$$\textbf{(BP2)} \quad \max \qquad F(\vec{y}) \tag{16}$$

$$\text{subject to} \qquad \sum_{i=1}^{m} \sum_{j=1}^{c} y_{ij} \leq k \tag{17}$$

$$0 \leq y_{ij} \leq 1, \quad \forall i \in I, \forall j \in J, \tag{18}$$

$$y_{ij} \in \{0, 1\}, \quad \forall i \in I, \forall j \in J. \tag{19}$$

We input BP2 and $\vec{y}^*$ to the *original* `PIPAGE` algorithm and run it iteratively until we obtain the integral vector, denoted by $\vec{y}^{\#}$. The integral solution $\vec{y}^{\#}$ provides the desired solution to MCMC—if $y_{ij}^{\#} = 1$, then monitoring node v_i is chosen to verify on channel j. We provide the formal description of DRS, denoted `DRS`, in Algo. 4. We also present in Algo. 5 DRA, denoted `DRA`, using `DRS` as its subroutine.

Algorithm 4 `Deterministic Rounding Scheme (DRS)`

```
1: y⃗' ← ỹ⃗
2: while (1) do
3:    y⃗* ← MOD-PIPAGE(BP1, y⃗')
4:    if y⃗* = y⃗' then
5:       break;
6:    else
7:       y⃗' ← y⃗*
8:    end if
9: end while
10: y⃗# ← y⃗*
11: while y⃗# is non-integral do
12:    y⃗# ← PIPAGE(BP2, y⃗#)
13: end while
14: return y⃗#
```

Algorithm 5 `Deterministic Rounding Algorithm (DRA)`

```
1: Formulate ILP-MC from a given MCMC
2: Transform ILP-MC into LP-MC by removing the integral constraint
3: Obtain the optimal solution ỹ⃗ = (ỹ_ij : i ∈ I, j ∈ J) to LP-MC (using one of existing LP solvers)
4: y⃗# = DRS(ỹ⃗)
5: return y⃗#
```

We now show the feasibility and performance of the solution from `DRA`. Before that, we first prove the following two lemmas.

Lemma 8. *It holds that* $\sum_{j=1}^{c} y_{ij}^* = \sum_{j=1}^{c} \tilde{y}_{ij},\ \forall i \in I$.

Proof. Observe that when `MOD-PIPAGE` runs, a chosen path always consists of two edges, which are connected to a common vertex in P_1. If the value of one edge is increased by ϵ, that of the other is decreased by ϵ. Hence, for every vertex $i \in P_1$, $\sum_{j=1}^{c} \tilde{y}_{ij}$ would remain same after a single execution of `MOD-PIPAGE`. Thus, it follows that $\sum_{j=1}^{c} y_{ij}^* = \sum_{j=1}^{c} \tilde{y}_{ij},\ \forall i \in I$. ■

Lemma 9. *$\vec{y}^*$ is a fractional solution to BP2.*

Proof. Since $\tilde{\vec{y}}$ is the optimal solution to LP-MC, $\tilde{\vec{y}}$ must satisfy the total budget constraint and hence we have that $\sum_{i=1}^{m} \sum_{j=1}^{c} \tilde{y}_{ij} \leq k$. Therefore, it follows from Lemma 8 that $\sum_{i=1}^{m} \sum_{j=1}^{c} y_{ij}^* = \sum_{i=1}^{m} \sum_{j=1}^{c} \tilde{y}_{ij} \leq k$, satisfying (17). As mentioned earlier in this subsection, `PIPAGE` takes a fractional solution as its input and outputs a new fractional solution. `MOD-PIPAGE` uses the same method to increment or decrement edge weights and therefore preserves this property of `PIPAGE`. Hence, $0 \leq y_{ij}^* \leq 1,\ \forall i \in I,\ \forall j \in J$, which satisfies (18). ■

We now show the feasibility of the solution of `DRA` in the following lemma.

Lemma 10. *`DRA` produces a feasible solution $\vec{y}^{\#}$ to ILP-MC.*

Proof. From Lemma 9 and the fact that `DRS` in the second phase runs `PIPAGE` until it gets an integral vector, we conclude that $\vec{y}^{\#}$ satisfies (3) and (7). To prove the lemma, we only need to show that $\sum_{j=1}^{c} y_{ij}^{\#} \leq 1,\ \forall i \in I$. Recall that

for $\vec{y}^*$, for every $i \in I$, there should be at most one $j \in J$ such that y^*_{ij} is non-integral. Hence, we have two possible cases for each i: the first case is that all y^*_{ij} are integral and the second case is that only one y^*_{ij} is non-integral. For every i_1 having the first case, $y^{\#}_{i_1 j} = y^*_{i_1 j}$, $\forall j \in J$ since integral edges are not modified, and therefore $\sum_{j=1}^{c} y^{\#}_{i_1 j} = \sum_{j=1}^{c} y^*_{i_1 j}$. Also, $\sum_{j=1}^{c} y^*_{ij} = \sum_{j=1}^{c} \tilde{y}_{ij}$ (by Lemma 8) ≤ 1, $\forall i \in I$. Thus, we get that $\sum_{j=1}^{c} y^{\#}_{i_1 j} \leq 1$. On the other hand, for every i_2 having the second case, $y^*_{i_2 j_1}$ for some $j_1 \in J$ is non-integral. Then, $y^*_{i_2 j} = 0$, $\forall j\ (\neq j_1) \in J$ since otherwise, for some $j_2\ (\neq j_1) \in J$, $y^*_{i_2 j_2} = 1$, and thus $\sum_{j=1}^{c} y^*_{i_2 j} > 1$. This cannot be true as just shown above within this proof. After running the second phase of DRS, we have rounded all non-integral edges. Hence, $y^{\#}_{i_2 j_1} = 0$ or 1. Thus, we get that $\sum_{j=1}^{c} y^{\#}_{i_2 j}$ is either 0 or 1, i.e., $\sum_{j=1}^{c} y^{\#}_{i_2 j} \leq 1$. Combining the two cases, $\sum_{j=1}^{c} y^{\#}_{ij} \leq 1$, $\forall i \in I$. ■

The following lemma is important to prove the performance of DRA.

Lemma 11. *$F(\vec{y}') \geq F(\vec{y})$, where $\vec{y}$ and $\vec{y}'$ denote the input and output vectors of PIPAGE (or MOD-PIPAGE).*

PROOF. Observe that for any fractional solution and any chosen path R, the function $F\big(\vec{y}(\epsilon, R)\big)$ is of the form $a_2\epsilon^2 + a_1\epsilon + a_0$, where $a_2 \geq 0$. Hence, $F\big(\vec{y}(\epsilon, R)\big)$, treated as a function of ϵ, is convex and thus attains the maximum at an endpoint of the interval $[-\epsilon_1, \epsilon_2]$. Since $\vec{y}' = \vec{y}(-\epsilon_1, R)$ or $\vec{y}(\epsilon_2, R)$, it follows that $F(\vec{y}') \geq F(\vec{y})$. This proof holds for both BP1 (MOD-PIPAGE) and BP2 (PIPAGE) since they differ only in the way the path R is chosen; they are identical in how the edge weights are updated. ■

We now show the performance of DRA in the following lemma.

Lemma 12. *The solution quality of DRA is at least $1 - (1 - 1/m)^m$ times the optimal value of ILP-MC.*

PROOF. Define $\{\tilde{x}_l, \tilde{y}_{ij} : l = 1, \ldots, n,\ i = 1, \ldots, m,\ j = 1, \ldots, c\}$ and $\tilde{z}$ to be the optimal solution to LP-MC and its optimal value, respectively. Let $x_l = \max\{y^{\#}_{ij} : u_l \in S_{ij}\}$. Then we have the following equation:

$$
\begin{aligned}
\sum_{l=1}^{n} x_l &= \sum_{l=1}^{n} \big(1 - \prod_{i,j:\ u_l \in S_{ij}} (1 - y^{\#}_{ij})\big) \\
&\geq \sum_{l=1}^{n} \big(1 - \prod_{i,j:\ u_l \in S_{ij}} (1 - y^{*}_{ij})\big) \quad \text{(by Lemma 11)} \\
&\geq \sum_{l=1}^{n} \big(1 - \prod_{i,j:\ u_l \in S_{ij}} (1 - \tilde{y}_{ij})\big) \quad \text{(by Lemma 11)} \\
&\geq \big(1 - (1 - 1/m)^m\big) \sum_{l=1}^{n} \min\{1, \sum_{i,j:\ u_l \in S_{ij}} \tilde{y}_{ij}\} \\
&\qquad \text{(from Lemma 6 and } m \geq p) \\
&= \big(1 - (1 - 1/m)^m\big) \sum_{l=1}^{n} \tilde{x}_l.
\end{aligned}
$$

The last equality is true since $\tilde{x}_l = \min\{1, \sum_{i,j:\ u_l \in S_{ij}} \tilde{y}_{ij}\}$, which has been shown in the proof of Lemma 7. Knowing that the optimal value $\tilde{z}$ of LP-MC is an upper bound of that of ILP-MC, concludes the proof. ■

Finally, we arrive at the following proposition about the performance of DRA.

Proposition 3. *DRA deterministically achieves the best approximation ratio $1 - 1/e$ unless $P = NP$.*

PROOF. It simply follows from Lemma 12 and that $1 - (1 - 1/m)^m > 1 - 1/e$. ■

As in PRA, DRA's solution quality is $1 - 1/e$ of the optimal for the asymptotic case of $m \to \infty$. For practical deployments, with a finite number of monitoring nodes, DRA deterministically achieves a better worst-case performance than $1 - 1/e$ times the quality relative to the optimal solution.

6. COMPLEXITY ANALYSIS

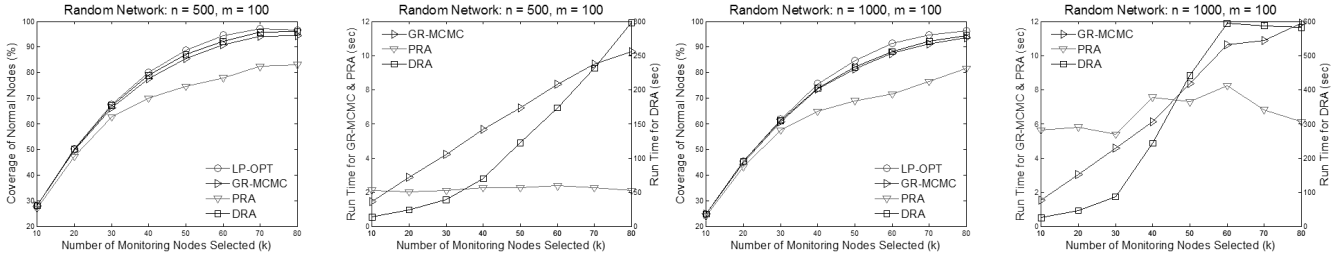
We first discuss the time complexity of the three algorithms: GR-MCMC, PRA, and DRA.

GR-MCMC picks the set of the maximum number of uncovered elements at each iteration and repeats it k times. Each iteration takes $O(mc)$. Therefore, GR-MCMC has time complexity of $O(kmc)$.

We next compute the complexity of PRA. Recall that PRA comprises three steps: 1) Formulate LP-MC; 2) Solve LP-MC using an LP solver; 3) Call PRS. In the first step, PRA builds from a given MCMC, an LP in the form: $\max(\vec{c}'\vec{x})$ subject to $A\vec{x} = \vec{b}$ and $\vec{x} \geq 0$. Building matrix A from the constraints (2)–(6) dominates the complexity in the first step. Hence, we focus on the the complexity of constructing A. We have $n+mc$ unknown variables in $\vec{x}$. In the constraint (2), we have n inequalities and thus it takes $O\big(n(n+mc)\big)$ to implement (2). Similarly, we can calculate the complexities for the other constraints (3)–(6), which are $O(n+mc)$, $O\big(m(n+mc)\big)$, and $O\big(mc(n+mc)\big)$, and $O\big(n(n+mc)\big)$, respectively. Therefore, the first step, setting up LP-MC, takes $O\big((n+mc)^2\big)$. Solving LP-MC in the second step is $O\big((n+mc)^3/\log(n+mc)\big)$, which is obtained by using the complexity of linear solver in [2]. In the third step, PRA calls its subroutine PRS, which in turn calls SAMPLING. The algorithm SAMPLING has complexity linear in the size of its input [11]. The algorithm PRS executes SAMPLING by first feeding an input of size $m+1$ in line 2 in Algo. 2 and then calls SAMPLING $O(m)$ times with inputs of size c, in lines 3–9 in Algo. 2. Hence, PRS takes $O(mc)$, which is the complexity of the final step of PRA. Thus, solving LP-MC in the second step dominates the complexity and thereby PRA has overall complexity $O\big((n+mc)^3/\log(n+mc)\big)$.

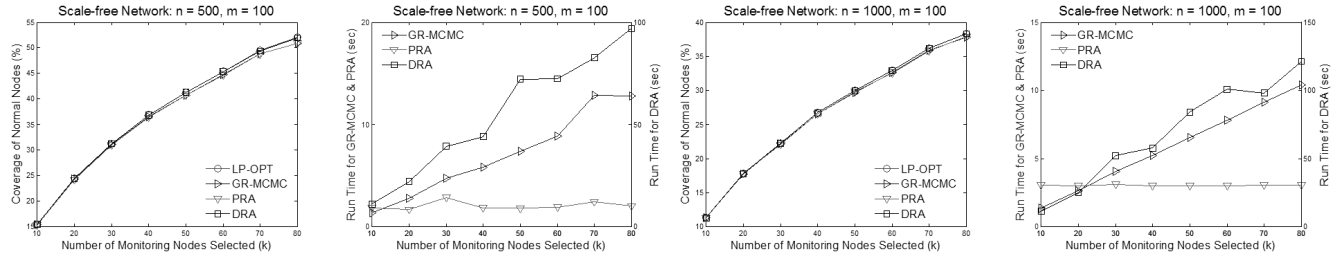
Finally, we calculate the complexity of DRA. Since DRA and PRA have the first two steps in common, we can use the result obtained above for PRA. In Algo. 4, DRS runs MOD-PIPAGE at most mc times and PIPAGE at most m times. Observe that both MOD-PIPAGE and PIPAGE take $O(nmc)$. Therefore, DRS has a complexity of $O\big(n(mc)^2\big)$. Thus, in DRA too, solving LP-MC is dominant in complexity and hence has overall complexity $O\big((n+mc)^3/\log(n+mc)\big)$, same as PRA.

We next discuss the communication complexity of the three algorithms. All three algorithms must be aware of $\mathcal{S}$, a collection of coverage-sets, which is global information. A central authority first sends queries to monitoring nodes, then each monitoring node replies with its coverage-sets to the central authority, and the central authority finally distributes to monitoring nodes its determination of which monitoring nodes and channels were picked. Therefore, $m+2$



(a) Coverage for $n = 500$ (b) Run time for $n = 500$ (c) Coverage for $n = 1000$ (d) Run time for $n = 1000$

Figure 2: Random network



(a) Coverage for $n = 500$ (b) Run time for $n = 500$ (c) Coverage for $n = 1000$ (d) Run time for $n = 1000$

Figure 3: Scale-free network

network-wide communications are required in total. However, for `GR-MCMC`, we can lower the communication cost to three network-wide communications and local communications by employing the approach used in MUNEN-MC for the single channel problem [12].

7. SIMULATION RESULTS

We evaluate the performance of the three proposed algorithms through simulations in random networks and scale-free networks. A *scale-free* network is a network, where a distribution $f(d)$ of nodes with degree d follows a power law, most commonly, of the form d^{-r}, where r is a constant. Thus, there is an exponentially decreasing number of nodes with high degree. Many empirically observed networks such as the world wide web appear to be scale-free.

For random networks, we randomly place normal nodes and monitoring nodes on a 1×1 square area. Receiving ranges of monitoring nodes are set to 0.15. For scale-free networks, they are generated with parameter r, $2 < r < 3$. There is no explicit receiving range in scale-free networks. Rather, the coverage of a monitoring node is determined by the degree of the node. We pick the m nodes with the highest degree as monitoring nodes, which is reasonable since such a choice can give a larger coverage than a random choice. We fix m to 100 and c to 4 in all simulations performed. Channels of normal nodes are randomly assigned. All the results are the averages over 30 iterations.

Figures 2(a), (b) show the results for $n = 500$ in random networks. In Fig. 2(a), LP-OPT denotes the optimal value of LP-MC, an upper bound of the optimal value of ILP-MC. Figure 2(a) shows that `DRA` has the highest coverage, `GR-MCMC` follows `DRA` with a small gap, and `PRA` has an inferior performance. `DRA`, `GR-MCMC`, and `PRA` have performance at least 97.5%, 96.2%, and 82% of LP-OPT, respectively. Figure 2(b) shows their running times. Notice that there are two different y axes. The left axis is for `GR-MCMC` and `PRA`, and the right one is for `DRA`. We can see that the run time of `GR-MCMC` linearly increases with k, as expected by the complexity analysis presented in Section 6. On the other hand, the results of `PRA` and `DRA` are surprising. Recall that `PRA` and `DRA` have in common their first two steps, which are formulating ILP-MC and solving LP-MC, and that both algorithms have the same asymptotic complexity since the second step dominates their complexities. However, Fig. 2(b) (Fig. 2(d) and Fig. 3(b), (d) also) shows quite a different result. `PRA` runs much faster than `DRA` and also faster than `GR-MCMC` for medium to large values of k. This result implies that in practice, the complexities of `PRA` and `DRA` are determined by their rounding schemes, `PRS` and `DRS`. Since they have the complexities $O(mc)$ and $O\big(n(mc)^2\big)$, respectively, their corresponding algorithms `PRA` and `DRA` have the results shown in Fig. 2(b). Further, note that the run time of `DRA` increases with k despite the fact that its complexity does not depend on k. This can be explained by the inference that when k grows, the number of non-integral components in the input to `DRS` increases, and `DRS` requires more iterations, thus the run time of `DRA` increases. Figure 2(c), (d) show the results for $n = 1000$ in random networks. We see similar results to those for $n = 500$. In both cases, the coverage achievable levels off at around 96%. This is due to a few unfortunately placed nodes that cannot be covered by any monitoring node.

Next, we evaluate the performance of the three algorithms in scale-free networks. Figures 3(a), (b) and Fig. 3(c), (d) show the results for $n = 500$ and $n = 1000$, respectively. First of all, we find in both settings that all the three algorithms have coverage very close to LP-OPT. Also, we can notice in both $n = 500$ and $n = 1000$ that `DRA` shows shorter

run time than that for corresponding n in random networks. These results suggest that scale-free networks provide all the three algorithms with more favorable collections of coverage-sets than random networks. Finally, we note that it is not correct to compare the coverage results of random networks with those of scale-free networks in the same settings since we have used different parameters to determine the coverage of a monitoring node—the receiving range for random networks and the parameter r for scale-free networks.

Summarizing our results, `DRA` shows the best performance very close to LP-OPT, however, its time complexity is high. Next, while `PRA` shows an inferior performance, it is computationally light. Finally, `GR-MCMC` performs quite close to LP-OPT and also has reasonable run-time performance. Hence, `GR-MCMC` is a good compromise between quality of solution and time efficiency. However, for critical security deployments, the network designer needs to guarantee the worst-case performance, a property that is provided by `PRA` and `DRA`.

8. CONCLUSIONS

In multi-channel multi-radio WMNs, we explore the problem of optimal selection of monitoring nodes and choice of channels for verifying the behavior of other network nodes. We mathematically formulate this problem, and show that obtaining the exact optimal solution is NP-hard. Thereafter, we present three algorithms, `GR-MCMC`, `PRA`, and `DRA`, to approximate the optimal solution. `GR-MCMC` is the intuitive extension of the existing greedy algorithm for the single channel case and achieves an approximation ratio of $\frac{1}{2}$, which is inferior to the best possible approximation ratio of $1-1/e$ under the assumption that $P \neq NP$. Then, we design the other two algorithms which use LP relaxation followed by rounding. The first algorithm, `PRA`, is probabilistic and attains the best approximation ratio on an average. On the other hand, `DRA` *deterministically* achieves this best approximation ratio. We then provide the time and communication complexities of the three algorithms. `GR-MCMC` has a time complexity linear in the number of monitoring nodes, the number of channels, and the maximum number of monitoring nodes that can be selected. `PRA` and `DRA` have the same worst-case time complexity, since the worst-case LP solver complexity dominates both. However, we find that in practice, `PRA` has complexity linear in the number of monitoring nodes and the number of channels, whereas `DRA` has complexity quadratic in the number of monitoring nodes and the number of channels and linear in the total number of normal nodes. Therefore, `PRA` performs better in terms of time complexity than `DRA` for the practical networks that we evaluate these solutions on. For communication complexity, all three algorithms require the number of monitoring nodes plus two times in the number of network-wide communications. However, for `GR-MCMC`, we can lower the communication cost to three network-wide communications and local communications by employing the approach used in MUNEN-MC for the single channel problem [12]. To evaluate how the three algorithms perform in practice, we conduct simulation in random networks and scale-free networks. The simulation results show that `GR-MCMC` is a good compromise between coverage and execution time. However, for critical security deployments, `PRA` and `DRA` are favored since they provide a superior performance guarantee in the worst case.

Our future work is on making the algorithms distributed while keeping the communication overhead manageable, and on designing solutions where a normal node is covered by multiple monitoring nodes for the scenario where the monitoring nodes can themselves be compromised.

9. ACKNOWLEDGMENTS

We thank Professor Gopal Pandurangan for pointing us to the literature on LP rounding techniques. We thank Professor Ness Shroff and Professor Issa Khalil for initial discussions in multi-channel WMNs using specialized monitoring nodes. This work has been supported in part by the National Science Foundation through awards CNS-0626830 and CNS-0831999.

10. REFERENCES

[1] A. Ageev and M. Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *Journal of Combinatorial Optimization*, 8:307–328, 2004.

[2] K. Anstreicher. Linear programming in O([n3/ln n]L) operations. *SIAM Journal on Optimization*, 9(4):803–812, 1999.

[3] C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. In *Proc. of Approximation, Randomization, and Combinatorial Optimization*, pp. 72–83, October 2004.

[4] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, July 1998.

[5] A. Haq, A. Naveed, and S. Kanhere. Securing channel assignment in multi-radio multi-channel wireless mesh networks. In *Proc. of IEEE WCNC*, pp. 3111–3116, March 2007.

[6] D. Hochbaum. *Approximation Algorithm for NP-Hard Problems*. PWS Publishing Company, Massachusetts, 1997, ch. 3.9.

[7] I. Khalil, S. Bagchi, and N. Shroff. LITEWORP: A lightweight countermeasure for the wormhole attack in multi-hop wireless networks. In *Proc. of IEEE/IFIP DSN*, pp. 612–621, 28 June – 1 July 2005.

[8] P. Kyasanur and N. Vaidya. Detection and handling of MAC layer misbehavior in wireless networks. In *Proc. of IEEE DSN*, pp. 173–182, June 2003.

[9] A. Naveed and S. Kanhere. Security vulnerabilities in channel assignment of multi-radio multi-channel wireless mesh networks. In *Proc. of IEEE GLOBECOM*, pp. 1–5, November 2006.

[10] N. B. Salem and J.-P. Hubaux. Securing wireless mesh networks. *IEEE Wireless Communications*, 13(2):50–55, 2006.

[11] A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *FOCS*, pp. 588–597, October 2001.

[12] D. Subhadrabandhu, S. Sarkar, and F. Anjum. A framework for misuse detection in ad hoc networks—Part I. *IEEE JSAC*, 24(2):274–289, Feburary 2006.

[13] Y. Yang, S. Zhu, and G. Cao. Improving sensor network immunity under worm attacks: A software diversity approach. In *Proc. of ACM MobiHoc*, pp. 149–158, May 2008.