

MOBILITY IN MOBILE SENSOR NETWORKS:
A STUDY OF SENSING PERFORMANCE AND PRIVACY

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Yu Tak Ma

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2010

Purdue University

West Lafayette, Indiana

To my supportive and always caring parents,
and my beloved wife Janet Wai Han

ACKNOWLEDGMENTS

In completing the work reported in this dissertation, I am most grateful to my advisor, Professor David K. Y. Yau, who has given me continuous support and guidance throughout the past five years, and led me to Purdue University in the first place.

I would like to thank Professors Nung Kwan Yip, Vernon Rego, Charles Killian, and Doctor Nageswara S. Rao for their time and efforts serving on my Ph.D advisory and examination committees, and for giving me valuable advice in my research and in writing this dissertation. I especially wish to thank Professor Nung Kwan Yip, with whom I collaborated on several research projects. The collaboration has enriched the analytical aspects of my work.

My research was supported in part by the Mathematics of Complex, Distributed, Interconnected Systems Program in the U.S. Department of Energy Office of Advanced Computing Research; in part by SensorNet Program at Oak Ridge National Laboratory managed by UT-Battelle, LLC for the U.S. Department of Energy under contract number DE-AC05-00OR22725; in part by a Purdue Summer Research Grant; and in part by a Bilsland Dissertation Fellowship. I gratefully acknowledge the financial support of these sponsors.

Spending five years in a place is not a short time, and life has been more enjoyable because of my friends here. I would like to thank all of my friends at Purdue, including Jren-Chit Chin, Lixia Liu, and especially Wan-ju Li who offered much help in the past four years which made my life a lot easier.

Last, but not least, I must thank everyone in my family. Especially my parents who gave me support and encouragement for pursuing my doctoral degree abroad, and my wife who gave me support, encouragement, faith, and trust, and experienced with me the ups and downs of my studies.

TABLE OF CONTENTS

	Page
LIST OF TABLES	vi
LIST OF FIGURES	vii
ABSTRACT	xi
1 Introduction	1
1.1 Problem Statement	1
1.1.1 Mobility algorithms for urban sensing	2
1.1.2 Privacy concerns for published mobility traces	3
1.2 Contributions	4
1.2.1 Mobility Algorithms for Urban Sensing	4
1.2.2 Privacy concerns for published mobility traces	5
1.3 Dissertation Organization	6
2 Mobility Algorithms for Mobile Sensor Networks	8
2.1 Problem Formulation	8
2.2 Related Work	9
2.3 Stochastic Mobility Algorithms	11
2.3.1 Stateful — Weighted Random Waypoint	11
2.3.2 Stateless — Markov chain	29
2.4 Evaluation	39
2.4.1 Performance of the WRW Mobility Algorithm	39
2.4.2 Performance of Different Coordination Approaches among Multiple Sensors	43
2.4.3 Performance of the Stateless Mobility Algorithm	48
2.5 Summary	59
3 Quality of Event Monitoring by Mobility Algorithms	60
3.1 Problem Formulation	60
3.1.1 Utility Functions	62
3.1.2 Definition of QoM	65
3.2 Related Work	66
3.3 QoM of Periodic Schedules	67
3.3.1 Single-PoI Analysis of QoM	67
3.3.2 Step Utility Function	70
3.3.3 General Utility Function	75
3.3.4 Implications of Theoretical Results	77

	Page
3.3.5 Discussions on Multiple Sensors	81
3.4 Evaluation	85
3.4.1 QoM of Periodic Schedule	86
3.4.2 QoM of WRW-aLP	96
3.5 Summary	105
4 Privacy Study	107
4.1 System Model	107
4.2 Related Work	109
4.3 Adversary Strategies	111
4.3.1 Passive Adversary	111
4.3.2 Active Adversary	115
4.4 Evaluation	115
4.4.1 Characteristics of Mobility Traces	117
4.4.2 Passive Adversary	124
4.4.3 Results for Active Adversary	130
4.5 Summary	140
5 Summary	141
5.1 Future Work	142
LIST OF REFERENCES	144
VITA	149

LIST OF TABLES

Table	Page
2.1 Matching performance of WRW-aLP as a function of pause time. . . .	43
2.2 Fraction of time sensor moves under WRW-aLP, for different maximum sensor speeds v and pause time parameters P	43
2.3 Coverage performance of stateless mobility algorithm.	52
2.4 Exposure performance of the stateless mobility algorithm.	52
2.5 Cost of the transition matrix of the stateless mobility algorithm computed using the adaptive algorithm and the perturbed algorithm.	54
2.6 Achieved performance using different ratios in the cost function. . . .	56
3.1 Maximum available information for capture, averaged over all the simulated events.	86
4.1 Basic statistics of the real traces.	118
4.2 Attack performance of an active adversary who is one of the participants.	132
4.3 Attack performance of a static adversary.	135

LIST OF FIGURES

Figure	Page
2.1 Specification of WRW-aLP algorithm.	16
2.2 Specification of map division algorithm.	20
2.3 Examples illustrating the output of the map division algorithms using the San Francisco city map for a 4-sensor case.	21
2.4 Map of residential area in San Francisco, its threat profile, and the matching performance achieved by progressive variants of WRW.	41
2.5 Rate of growth of global undercoverage time using WRW-aLP.	42
2.6 Maps of the cities under surveillance and their corresponding threat profile.	42
2.7 Tradeoff between matching and fairness by the pause time parameter of WRW-aLP.	43
2.8 Percentage deviation of WRW-aLP global coverage profile from given threat profile for different multi-sensor coordination approaches.	44
2.9 Unfairness of WRW-aLP global coverage profile from given threat profile for different multi-sensor coordination approaches.	45
2.10 Effective coverage of WRW-aLP global coverage profile from given threat profile for different multi-sensor coordination approaches.	46
2.11 Pause fraction of WRW-aLP global coverage profile from given threat profile for different multi-sensor coordination approaches.	47
2.12 Target per-PoI shares of coverage times in four simulation topologies.	50
2.13 CDFs of achieved cost by different steepest descent algorithms.	51
2.14 Cost as a function of iteration in the steepest descent algorithm.	53
2.15 Cost of the basic algorithm as a function of iteration number.	55
2.16 Performance of the basic algorithm and the perturbed algorithm as a function of iteration number.	55
2.17 Performance of the stateless mobility algorithm as a function of iteration number.	57

Figure	Page
2.18 Performance of the stateless mobility algorithm as a function of iteration number.	58
2.19 Coverage and exposure performance of the algorithm as a function of iteration number.	58
3.1 Illustration of event dynamics at a PoI.	61
3.2 Utility functions concerned.	63
3.3 Utility of radiation measurement as a function of window size.	64
3.4 Illustration of event and sensor dynamics at a PoI.	70
3.5 Illustration of sensor dynamics versus event dynamics for step utility function.	73
3.6 QoM for events with exponential utility function and exponential staying time.	80
3.7 Geometry of schedule and sensor dynamics at a PoI using multiple sensors.	82
3.8 Relation between event dynamics and the multisensor coverage pattern.	83
3.9 QoM for the step utility function with varying number of mobile sensors.	83
3.10 QoM for the delayed-step utility function with varying number of mobile sensors.	85
3.11 Achieved QoM for staying events with step utility function.	87
3.12 Achieved QoM for staying events with exponential utility function.	88
3.13 Achieved QoM for staying events with linear utility function.	90
3.14 Achieved QoM for staying events with delayed-step utility function.	91
3.15 Achieved QoM for staying events with S-shaped utility function.	92
3.16 QoM for staying events with step utility function using multiple sensors.	94
3.17 QoM for staying events with exponential utility function using multiple sensors.	94
3.18 QoM for staying events with linear utility function using multiple sensors.	95
3.19 QoM for staying events with S-shaped utility function using multiple sensors.	96
3.20 QoM for staying events with delayed-step utility function using multiple sensors.	97

Figure	Page
3.21 The three city maps and their corresponding threat profiles.	97
3.22 Per captured event utilities of WRW-aLP as a function of the maximum sensor speed.	98
3.23 QoM performance of WRW-aLP with multiple sensors.	99
3.24 The ring topology.	100
3.25 Utility of stochastic mobility algorithms with exponential utility function.	101
3.26 Utility of stochastic mobility algorithms with step utility function. . . .	103
3.27 Utility of stochastic mobility algorithms with linear utility function. . .	104
3.28 Utility of stochastic mobility algorithms with delayed-step utility function.	104
3.29 Utility of stochastic mobility algorithms with S-shaped utility function.	105
4.1 Specification of active adversary attack algorithm.	116
4.2 Distribution of correlations between traces of the same set.	119
4.3 Autocorrelation of each trace for different sets of traces as a function of the time shift s	120
4.4 Order- n complexity of different sets of traces as a function of order. . .	122
4.5 Average and minimum distances between each node pair.	122
4.6 Performance of privacy attack of passive adversary with correct assumption on noise.	125
4.7 Performance of privacy attack of passive adversary with possible incorrect assumption on noise.	127
4.8 Performance of privacy attack of passive adversary when location inference is needed (cab traces).	129
4.9 Performance of privacy attack of passive adversary when location inference is needed (bus traces and random waypoint traces).	130
4.10 Average k -anonymity as a function of attack time when the adversary is one of the mobile nodes.	131
4.11 Distribution of total number of meeting time to identify a victim when the adversary is one of participants.	133
4.12 Distribution of number of non-consecutive meetings to identify a victim when the adversary is one of participants.	133

Figure	Page
4.13 Distribution of time to identify a victim when the adversary is one of participants.	134
4.14 k -anonymity of the victim as observed by a static adversary as a function of attack time.	136
4.15 Distribution of number of encounters to identify a victim when the adversary is static.	137
4.16 Distribution of time (from beginning) to identify a victim when the adversary is static.	138
4.17 k -anonymity of the victim as observed by the adversary as a function of attack time, when the adversary is mobile following a pre-determined path.	139

ABSTRACT

Ma, Yu Tak. Ph.D., Purdue University, December, 2010. Mobility in Mobile Sensor Networks: A Study of Sensing Performance and Privacy. Major Professor: David K. Y. Yau.

Recent advances in sensor technologies have made sensors more economical, power efficient, and portable to be mounted onto handheld devices for monitoring different environmental factors, and made mobile sensor networks possible.

When it is financially infeasible to deploy enough or an excessive number of sensors, while the environmental factors they monitor are critical for public health and safety, such as chemical or radiation monitoring, we deploy mobile sensors that move under our control. We also decide the best mobility strategy to achieve the desired goals. We propose and analyse mobility strategies that give a well-balanced performance for various goals which may be antagonistic. We notice that for a stochastic mobility algorithm, pausing at a location is well-justified to achieve better quality in event monitoring and a closer match with the expected monitoring time of a location by the sensor. We also notice that the quality of event monitoring at a location may not be proportional to the time the sensors spend at the location.

In other cases when it is economical to deploy an excessive number of sensors to monitor the environment by attaching them to electronic devices owned by the public, traces of mobile nodes are collected to help design and analyse of such systems and evaluate the expected performance before deployment. We are interested in studying privacy leakage through trace publication. Although published traces have their identity being replaced consistently with random IDs, movements of mobile nodes can be openly observed by others, or they may be learned through web blogs, status in social networks, and causal conversations, etc. It is then possible for an attacker to learn

the whole movement history of the participants, breaching the privacy protection. We study comprehensively attack strategies both analytically and experimentally using real and synthetic traces. We observe that with high probability an adversary can identify participants in the trace set with the current scale of trace collection and publication.

1 INTRODUCTION

Mobile sensor networks have become an active area of research in recent years. The excitement is brought by advances in sensor technologies, which have led to greatly improved performance and capabilities of the sensors in many aspects. In particular, sensors have been made more precise for event detection, more power efficient in operations, and more economical in production. They have also become smaller in size, and can be attached to various handheld electronic devices carried by people and other mobile agents. Exciting applications have been developed, including urban sensing of radiation, biological agents, and chemicals for people protection [1,2], traffic monitoring [3], and road surface condition sensing [4].

1.1 Problem Statement

Different applications impose different requirements and constraints on the design and deployment of mobile sensor networks. For instance, radiation, biological, and chemical sensors that give precise measurements remain relatively expensive, making them infeasible to be extensively deployed in a city setting. To reduce the sensing cost, these sensors can be carried by vehicles and other mobile entities on patrol in deployment areas for the purpose of threat detection. The mobility algorithms utilized by these mobile nodes must then be designed to meet the objectives of the subject applications. On the other hand, sensors useful for applications such as traffic monitoring and road surface sensing are less complicated than specialized sensors for radiational, biological, and chemical plumes. These more general purpose and less expensive sensors may be carried around by ordinary users in everyday life.

In this section, we introduce two problems of mobile sensor networks studied in this dissertation.

1.1.1 Mobility algorithms for urban sensing

The first problem we study is the analysis of stochastic mobility algorithms for sensors to achieve certain event monitoring objectives. From past sensor network deployments for detecting chemical and radiation threats [5,6], we have learned the following lessons.

- *The need for resource management.* Radiation or chemical sensors with high precision are expensive. Hence, it is not financially feasible to deploy enough sensors to cover expansive areas such as whole urban cities. With static sensors, different heuristic deployment strategies have been employed to determine the best locations for their placement [5,6], such that user- or application-specific objectives in coverage and connectivity can be satisfied.
- *The need for human protection.* One major objective of urban-sensing applications is human life protection [6]. In particular, the failure to deploy sensors at optimal locations for countering chemical or radiation threats can lead to unnecessary compromise of the public's safety. Hence, placement of sensors in a people-centric manner is highly desirable.
- *The need for event uncertainty reduction.* Because of the nature of the events being detected, the limitations of measurement devices being deployed, or inherent natural phenomena such as background noise, a single measurement reported by the sensors may not give high confidence information about the targeted events. For instance, measurements for radiation detection are inherently probabilistic with high variance [7]. High confidence in the event monitoring is only possible after taking measurements over a non-negligible period of time, leading to the important temporal dimension of sensing.

Against the above background, *first* we will propose and analyze stochastic mobility algorithms to handle the first two requirements. We will use mobile sensors such that every relevant location in the network area can be covered over time. Mean-

while, the portion of time sensors spend at each location is allocated to each location is designed to be proportional to the *threat-level* associated with the location. Intuitively, the threat-level of a location indicates the level of damage if the case that a threat materializes there but is left undetected; it could be quantified as the human casualties or financial loss resulting from the threat.

To tackle the third requirement, *next* we analyze how the quality of event monitoring is affected by the proportional share as well as the sharing granularity of a sensor's coverage schedule at each location. Our analysis will account for different real-world dynamics of events and types of events as characterized by a *utility function* for the temporal dimension of sensing each event type.

1.1.2 Privacy concerns for published mobility traces

Common sensors that are inexpensive and highly portable can be carried by everyday people and/or vehicles to help monitor possible threats in an urban area. Clearly, the mobility patterns of potential real-world participants in these networks, including their correlations and interactions with each other, will have large impact on the network performance (e.g., coverage and connectivity in the collaborative sensing network). Researchers have also found that existing synthetic movement models of mobile entities, such as pedestrians and different kinds of vehicles, though attractive for their low production cost and high repeatability, generally fail to capture essential behaviors of real users. Therefore, the use of synthetic traces in network design can lead to wrong conclusions about network performance (e.g., routing efficiency) in reality [8]. Hence, there are increasing efforts to trace the locations of real users, leading to the public availabilities of many such traces through either consolidated data portals such as Cawdad [9] or websites set up by individual research groups [10].

In order to protect the privacy of participants in real user traces, the true identity of each participant is often replaced by a consistent, unique, and random identifier that is not correlated in any way with the true user identity. Moreover, the precision

of the traces in the spatial and temporal domains can often be reduced by *cloaking* techniques such as reducing the resolution of the recorded data or introducing noise deliberately in the data. It is not clear, however, if these “anonymization” and cloaking techniques are sufficient to protect the privacy of the participants. This is because movements or whereabouts of participants in public spaces can be openly observed by others through chance or engineered meeting opportunities. Similar location or movement information can also be learned indirectly from conversations, news articles, publicly viewable webcams, online social networks, tweets, or web blogs, though the inference could be noisy. By gathering a few such (possibly rough) snapshots of a participant’s location over time, which we term as *side information*, an adversary may be able to identify (either uniquely or with high probability) the participant’s trace from a set of anonymous traces. Hence, the complete whereabouts of the participant (the *victim*) over an extended time duration will be revealed to the adversary.

The second problem we study in this dissertation is a formal analysis of the above problem. We will provide comprehensive quantitative analysis and experimental results under different mobility patterns and attack strategies.

1.2 Contributions

1.2.1 Mobility Algorithms for Urban Sensing

Concerning the problem of designing mobility algorithms for sensors in area and event monitoring, our contributions are as follows.

- To the best of our knowledge, ours is the first effort to investigate general threat-based coverage by mobile sensors encompassing both stochastic and deterministic algorithms.
- We identify *matching* and *fairness* as two major performance criteria in evaluating the effectiveness of coverage. We show that the two performance measures are in general antagonistic, and discuss their tradeoffs.

- We present the design of a stateful mobility algorithm for sensors to achieve effective matching and fairness simultaneously under realistic deployment scenarios. The algorithm provides a tunable parameter to control the tradeoff between matching and fairness.

We also present the design of a stateless mobility algorithm, and the use of a steepest descent approach to determine the optimal parameters in the algorithm. We expose the presence of possibly numerous local optima in the search space due to the interactions of multiple objectives under a large number of control parameters. We show how controlled noise can be introduced into the search strategy so that the algorithm can avoid being stuck at a local optimum and arrive at a global optimum with high probability.

- We analyze the quality of monitoring (QoM) for different types of events using mobile sensors. We provide extensive analysis to study how the granularity of sensor coverage and different levels of proportional share of sensor resource at a cell affect the QoM as a function of the event dynamics and type of events. In contrast to prior results, we show that whether increased mobility is beneficial depends critically on how information is gained as the event monitoring time increases.

1.2.2 Privacy concerns for published mobility traces

Concerning the privacy issues for publishing mobility traces, our contributions are two-fold.

- We provide extensive analysis both theoretically and experimentally to demonstrate that with the practice of recording and publishing anonymous location traces of reduced spatial and temporal granularity of real users, the concern exists that an adversary could identify the traces of one or more victims in the published data set with high probability, by learning a small amount of side

information about the participants. In particular, we present comprehensive attack strategies available to the adversary when it collects information about a victim’s movement either through direct observations or indirect information sources, and show that these attacks are effective in breaching privacy. We also provide a mathematical framework to show the optimality of specific attack strategies in that they utilize all the available information in the most effective way.

- We give comprehensive experimental analysis to show the differences between synthetic and real traces from the perspective of the privacy problem. Despite generated from the same basic statistics (e.g., size of the network and average speeds of the mobile nodes) of the real traces, the synthetic traces may behave quite differently from the real traces. Their different characteristics result in different performance of various privacy attacks. For instance, mobile nodes in the synthetic traces are more sparsely distributed in the network. This leads to easier de-anonymization of synthetic traces when the adversary attacks by collecting side information passively, but the de-anonymization may take a longer time if the adversary attacks by observing the participants directly. The different characteristics of the traces may also reduce the effectiveness of a privacy protection measure if it is designed and verified using the synthetic traces.

1.3 Dissertation Organization

This dissertation is organized as follows. In Chapter 2, we study the use of stochastic mobility algorithms to achieve given user-specific or application-specific goals in sensor networks. We will evaluate key coverage properties of the mobility algorithms, including the ability of the coverage to match a given threat profile of the network, and the exposure of different locations to possible threats during the monitoring. We will analyze the performance of both a stateful and a stateless algorithms analytically and experimentally. In Chapter 3, we analyze the quality of event monitoring

by mobile sensors for different types of events. We show how different proportional shares and fairness granularities may affect the monitoring performance at different network locations. This will lead to a basic understanding of the impact of mobility in monitoring tasks. In Chapter 4, we study the problem of privacy breaches in published mobility traces. We analyze effective attack strategies by an adversary to identify a victim's trace given side information about the victim obtained from other sources. We mathematically prove the optimality of certain attack strategies in that they can make full use of the available information in the attack. We then present a comprehensive experimental study on the characteristics of various synthetic and real mobility traces, and discuss their implications in the privacy attack problem. We summarize the dissertation and give future directions of research in Chapter 5.

2 MOBILITY ALGORITHMS FOR MOBILE SENSOR NETWORKS

In this chapter we study the problem of using mobile sensors to provide protection by detecting threats in a network area. To accomplish the task, the sensors use a mobility algorithm to direct their movements, which aims at optimizing the user-specific or application-specific goals. We first propose and explain the rationale of two different mobility algorithms (one is stateful [11] while the other one is stateless [12],) and then evaluate their performance analytically and experimentally in terms of coverage properties.

2.1 Problem Formulation

We consider the monitor of a network area, which we call the *map*, for a given threat by one or more mobile sensors. For simplicity, we assume that the map is a two-dimensional rectangular space of dimensions $x \times y$, where x and y are in distance units. The map is partitioned into an $m \times n$ array of cells, each of dimensions $S \times S$ (in distance units), where S divides both m and n . The cells are enumerated by their unique integer ids, $0, 1, \dots$, in top-to-bottom row order, and left-to-right column order within each row.

The distribution of threat in the map area is specified by a *threat profile*, denoted by Φ . The threat level of a cell, say i , is given by $\Phi(i)$, and quantifies the risk of not covering the cell relative to the other cells. The threat profile is determined according to the application, namely the kind of threat, and any relevant meteorological, environmental, and human factor. In addition, we allow certain cells to be marked as *inaccessible*, meaning that a sensor can neither monitor nor travel over such a cell due to physical limitations or policy decisions, which are excluded from the threat profile. Mathematically, Φ is a probability distribution, $0 \leq \Phi(i) \leq 1$ and $\sum_j \Phi(j) = 1$.

To address threat-based sensing, in solving the coverage problem, areas of higher threat level should receive greater attention in the form of proportionately higher coverage. The goal is to be achieved by a mobility algorithm that controls the movement of the sensors. As a sensor moves, it visits different cells. For the purpose of bookkeeping, we assume that a cell is *covered*, in the sense that any threat event presents in the cell is detected, whenever a sensor is inside the cell. By bookkeeping on a per-cell basis, the bookkeeping costs can be kept low, at the cost of possibly losing some precision in matching. However, the precision loss is small if the cell dimension S is comparable to the sensing range of the sensor. The fraction of time that a sensor, say l , spends in each cell up to time t is given by the sensor's *coverage profile*, denoted by Π_t^l . Specifically, $\Pi_t^l(i)$ gives the fraction of time that the sensor l spends in the cell i up to time t . Similar to Φ , Π_t is a probability distribution. When the context of the sensor is clear, we drop the superscript l for simplicity.

For simplicity, we assume that the threat profile is time invariant. In practice, it is clear that when the threat profile changes, we can use the current profile as a new input to the mobility algorithm, and the sensor's mobility will adapt accordingly. Threat profiles in real life are likely to depend on factors that are close to static because they change slowly – e.g., population, locations of strategic facilities, and seasonal changes of weather.

2.2 Related Work

Sensor coverage problem has received a lot of attention in the literature. In static sensor domain, [13,14] studied the minimum number of static sensor devices required to cover a surveillance area. Clouqueur *et al.* [15] studied algorithm to optimally place static sensors in a surveillance area to obtain best coverage that minimize the exposure in target intrusion monitoring. In environmental monitoring application, [6] solved the problem of optimal placement of static sensors to protect population against release of toxic substance including chemical, biological, and radiological substances.

Meteorological data were used in modeling dispersion and calculate population at risk.

The above-mentioned work rely on the use of static sensors to detect, identify and track event of interest, moving target, and spreading toxic substance. In recent years, research have also focused on using mobile sensors to perform similar tasks to gain robustness and achieve higher efficiency of the sensor network. For instance, Batalin *et al.* [16] have studied the use of static and mobile sensors in a hybrid sensor network to support high fidelity environmental monitoring application. Other researchers have also studied coverage of mobile sensors [17–19]. Nevertheless, our study is very different from their problems. Namely, sensor mobilities in those papers are limited to movement during deployment to maximize the coverage of the area of interest. Once the sensors have moved to the appropriate position, they remain static thereafter. Researches in [20, 21] concerned about finding the least covered path in the area, which is interpreted as appropriate places for extra sensors. However, the sensors' position remain static after deployment. Rao *et al.* [22, 23] addressed the terrain model acquisition problem by mobile robots in the formulation of convergent algorithms using visibility graphs. Rao [24] showed the importance of the connectivity of a space to determine the size of the robot team that is effective to explore the terrain, and proposed methods to coordinate members of the robot team for efficient exploration. However, the sensors do not need to monitor the terrain after the exploration.

For event detection, [25] studied the problem of using mobile sensors to capture transient events that appear and fade away. Their work differs from ours in the following fundamental aspects. First, they consider a collection of point of interests scattered along a simple closed curve whereas our work consider a rectangular surveillance area to be protected by a sensor network. Second, their work uses mobile sensors that move deterministically along the closed curve to visit each point of interest in sequence. Our work uses mobile sensors to patrol the surveillance area with stochastic movement to detect event of interest. Third, there is no accessibility constrain in their model as the mobile will move deterministically in the closed curve. In contrast,

we take the more generic approach where the mobile sensors will take accessibility constraint into account when planning a trip. Lastly, our mobility algorithms are adaptive to the needs of coverage (measured in terms of *exposure*) where the mobile sensors will provide coverage to the area that needs the most whereas theirs is not.

Hsu *et al.* have investigated a weighted version of random waypoint model denoted as weighted waypoint model WWP in [26]. Nonetheless, their work is fundamentally different from ours. In their paper, they are *modeling* physical movement of people in real world and evaluated their model by profiling students' movement in University of Southern California campus. Our work seeks for an *algorithm* that mobile sensor can use to effectively monitor a surveillance area and *not* to model physical movements. In general, mobility models that are widely studied in the literature are proposed to model the movement of physical entities such as human and vehicles. This chapter studies mobility algorithms that are designed to *guide* movement of mobile sensors such that the movement provide effective coverage to the surveillance area, given an arbitrary risk distribution.

2.3 Stochastic Mobility Algorithms

In this section we propose and analyse two different types of mobility algorithms to instruct the movement of the mobile nodes, such that user-specific or application-specific goals can be achieved.

2.3.1 Stateful — Weighted Random Waypoint

In this section, we develop stateful mobility algorithms for a mobile sensor to follow. We aim at the following desirable properties of the algorithm in our design:

- *Accurate.* The algorithm should give a close match with the expected resource allocation.

- *Fair*. The algorithm should be fair, i.e., no cells in the network area is exposed to threats for a prolonged period of time.
- *Stochastic*. The random movement makes it hard for an adversary to anticipate the sensor's movement and hence avoid detection.
- *Efficient*. The algorithm should have low space and time complexities, so that it can be efficiently executed on the mobile sensor.
- *Practical*. The algorithm should obey given *accessibility constraints* for the coverage area, such as avoiding movement over inaccessible areas. For example, a sensor carried on a terrestrial vehicle will not be able to enter sea areas in a geographical region.

As a starting point of our design, we use a *weighted random waypoint* (WRW) algorithm. The random waypoint formulation proposed in [27] has been used widely to *model* user/device movement in a mobile network, although there is a significant debate about whether the model is realistic or not. Notice that the concern of realism does not apply in our problem context, since our objective is not to model the movement of a mobile node, but to develop an *algorithm* for determining sensor movement. In our algorithm, a sensor moves in a sequence of *trips*. The t -th trip, $t = 0, 1, \dots$, starts at a (uniformly) random position in cell s_t and ends at a (uniformly) random position in cell d_t , the $(t+1)$ -st trip starts at the random position in cell $s_{t+1} = d_t$ and ends at a random position in cell d_{t+1} , and so on. We pick a random position with uniform probability inside a cell, because each cell is of non-negligible dimensions $S \times S$. For simplicity, we assume henceforth that when we say a trip starts/ends at a cell, it is understood that the trip starts/ends at a random position inside the cell. The movement from s_t to d_t occurs in a direct, straightline path at speed v_t ($v_t > 0$). Moreover, to be threat-aware, our algorithm considers the given threat profile when choosing a waypoint, and selects a cell i as the waypoint with probability Φ_i , which is the threat level of i .

The WRW algorithm is simple and probabilistic, thus meeting the third and fourth design objectives. Moreover, it attempts to achieve a coverage that matches the threat profile, by considering the profile in selecting the waypoints. The basic algorithm, however, fails to achieve an accurate match because it fails to consider the effects of a trip on covering the *intermediate* cells between the source and destination. For example, consider a map with a few high threat *hotspots*. In attempting to move between the hotspots to give them sufficient coverage, the sensor will also visit frequently all the cells between the hotspots, thereby overcovering the intermediate cells. To overcome the weaknesses of the basic algorithm, WRW can be used in conjunction with the following complementary features:

- **Adaptivity to prior coverage.** Because of the stochastic nature of the WRW algorithm, and the correlations between cells visited due to their physical positions, the algorithm’s actual coverage at any point in time may deviate from the given threat profile. To avoid such deviations from accumulating to an unacceptable level, we propose to use the sensor’s prior coverage as an input in selecting the next waypoint. Specifically, we compute the *undercoverage* of each cell, say i , as

$$\bar{C}_t(i) = \max\{0, \Phi(i) - \Pi_t(i)\},$$

where $\Pi_t(i)$ is the fraction of time that cell i was visited by the sensor up until the end of the t -th trip. Then, the probability that a candidate cell, say i , is chosen as the next waypoint d_{t+1} is proportional to $\bar{C}_t(i)$. Considering undercoverage as a selection criterion has the obvious advantage of increasing the probability to visit cells that have been neglected relative to their threat level, at the expense of cells that have received too much prior coverage.

- **Maximum trip length.** In this variation, we do not allow the distance of a trip to exceed a given parameter L (in distance units). Hence, in choosing the next waypoint d_{t+1} after the t -th trip, we constrain the candidate cells to be within a disc centered at d_t of radius L . The choice of the destination among the

restricted set of candidate cells occurs as in the basic algorithm. Limiting the trip length helps to decouple the intermediate cells visited from a set of high threat cells that require frequent visits. For example, consider two hotspots, say i and j , in a map. A suitable maximum trip length will force the sensor to consider more possible paths to move between i and j , thereby reducing the possibility of “warming up” the intermediate cells as a side effect.

- **Random pause time at the destination.** To raise the coverage of an undercovered cell, say i , in order to improve matching with the threat profile, the most efficient approach is for the sensor to stay in i for long enough time to correct the undercoverage. The approach is extremely efficient because it requires zero overhead of movement and there is no possibility of inadvertently changing the coverage of other cells due to the (now avoided) movement. However, by staying at the current cell longer, clearly the sensor will take longer time before it can return to a previously visited cell. Hence, fairness suffers, showing that there is an inherent tradeoff between improving matching efficiently/accurately and being fair. To enable a useful and controllable tradeoff between the matching and unfairness metrics, the sensor, on reaching the destination of a trip, will stay at the destination for a *pause time* p (in time units) before selecting the next waypoint. The time p is drawn randomly from a distribution determined by a pause time parameter denoted by P (in time units). Specifically, at the end of the t -th trip at destination cell i , $p \sim \mathbf{U}(0, \Omega_t(i))$, where

$$\Omega_t(i) = \frac{P \times \Phi_t(i)}{\sum_{j \in \mathcal{C}} \Phi_t(j)}$$

for the basic WRW algorithm, and

$$\Omega_t(i) = \frac{P \times \overline{\mathcal{C}}_t(i)}{\sum_{j \in \mathcal{C}} \overline{\mathcal{C}}_t(j)}$$

for the WRW variant that is adaptive to prior coverage, and \mathcal{C} is the set of cells that are candidates as the next waypoint.

Family of algorithms. Notice that the complementary features augmenting the WRW algorithm can be orthogonally combined, thereby offering a *family* of algorithms for threat-based mobile coverage. We will denote a particular augmented algorithm by $WRW-feat$, where $feat$ is a list of letters enumerating the augmentations in alphabetical order, and the letters a, L, and P, are for the “adaptivity to prior coverage”, “maximum trip length”, and “random pause time” features, respectively. For example, WRW-L denotes the WRW algorithm with the maximum trip length constraint, and WRW-aLP denotes the algorithm with all the three features enabled. The experimental results in Section 2.4 show that each feature contributes positively to accurate matching, and hence the WRW-aLP algorithm is the most powerful in the matching respect. Additionally, the pause time parameter in WRW-aLP enables a useful tradeoff between matching and fairness.

The WRW-aLP algorithm is specified in Fig. 2.1. In the specification, the **WRW-aLP** program takes as input the threat profile Φ , and the L and P parameters of the WRW-aLP algorithm. The function **SelectWaypoint** takes four input parameters, in which the parameter x_t is the current position of the sensor, and returns the destination and pause time of the next trip. The **Accessible** function (whose specification is not shown) checks if all the intermediate cells connecting a given pair of cells are accessible, and can be precomputed for each given pair. Either for-loop in **SelectWaypoint** has a complexity of $O(L_s^2)$, where $L_s = L/s$. Hence, WRW-aLP requires $O(L_s^2)$ computation after every trip of length $O(L)$. The space costs of storing either the map of cells or the precomputed **Accessible** function is $O(m \times n)$. Hence, WRW-aLP can handle given inaccessibility constraints and has an efficient implementation.

Multi-sensor Coordinations

When the network is large relative to the sensing range of a sensor, it is hardly sufficient to deploy a single sensor to monitor the whole area. Even though the sensor

```

1  SelectWaypoint ( $L, P, \bar{C}_t, x_t$ )
2    Initialize  $u \sim \mathbf{U}(0, 1)$ ,  $a := 0$ ,  $b := 0$ ,  $c := 0$ ;
3    For each cell  $i$  within range  $L$  of  $x_t$ 
4      If ( $\text{Accessible}(x_t, i)$  and  $\bar{C}_t(i) > 0$ )
5         $a := a + \bar{C}_t(i)$ ;
6        If ( $b < \bar{C}_t(i)$ )
7           $b := \bar{C}_t(i)$ ;
8        Endif
9      Endfor
10   For each cell  $i$  within range  $L$  of  $x_t$ 
11     If ( $\text{Accessible}(x_t, i)$  and  $\bar{C}_t(i) > 0$ )
12        $c := c + \bar{C}_t(i)/a$ ;
13       If ( $u < c$ )
14         pick  $x_{t+1}$  as random point inside  $i$  with uniform probability;
15          $p := (P \times \bar{C}_t(i))/(a \times b)$ ;
16       Endif
17     Endif
18   Endfor
19   return  $(x_{t+1}, p)$ ;

20 WRW-aLP( $\Phi, L, P$ )
21 While (true)
22    $(x_{t+1}, p) := \text{SelectWaypoint}(L, P, \bar{C}_t, x_t)$ ;
23   move to  $x_{t+1}$ ;
24   pause for  $p$  time;
25   update  $\bar{C}_t(i)$ ;
26 Endwhile

```

Figure 2.1. Specification of WRW-aLP algorithm.

is able to navigate the whole area and cover it eventually, the exposure time will be far from satisfactory. When multiple mobile sensors are deployed in the network, it may be important to coordinate their movements such that they could provide the optimal performance. In this section we consider four different coordination approaches. We first present the details of these approaches, and then evaluate their performance in Section 2.4.

The four approaches are listed as follows,

- *No coordination, NC.* In this approach the mobile sensors do not coordinate their movements with each other, and they take no measures to prevent duplication of effort in coverage. Each mobile sensor monitors the whole network area following the WRW-aLP algorithm using each own coverage history.
- *Global knowledge of coverage history, GK.* In this approach, the coverage history of every node is shared among all the mobile nodes, and the overall effective coverage after removing duplicated coverage by multiple mobile sensors is used as the coverage history for the WRW-aLP algorithm. We assume that there exists a mechanism to allow the sharing of coverage history among all of the mobile sensor nodes, such as the existence of an infrastructure. The goal here is not to study how the information can be shared among the nodes, but how effective it is to share such information among the mobile nodes.
- *Static division of sensor responsibility (no overlapping), MD.* In order to completely eliminate the redundancy between sensors, we sub-divide the whole covered area into *non-overlapping* subregions. One single sensor is dedicated to each sub-region and it follows its own WRW-aLP algorithm. Note that because of the non-overlapping constraints, in general, the total threat-levels in the sub-region will not be the same. The matching and unfairness of this approach rely heavily on the topology of the PoIs, the distribution of threats among the PoIs, and the manner in which the network is partitioned among the sensors. In particular, a sensor in one assigned subregion will not be able to compensate

for any undercoverage of PoIs in another subregion. Although the redundancy of coverage problem is eliminated in MD, the sensors' effective coverage is still not 100% due to travel overhead between the PoIs. Nevertheless, with proper division of the surveillance area, this travel overhead can be greatly reduced, and the effective coverage of MD may be significantly better than either NC or GK, thereby improving the information capture.

The algorithm to divide the network is specified in Fig. 2.2. An example illustrating the output of the algorithm is shown in Fig. 2.3(a). The goals of the algorithm are to divide the area into subregions of similar aggregate threat levels for good matching performance, while none of the subregions should overlap so that redundant sensing can be eliminated. Notice that the algorithm assumes that the graph consists of a single connected component. If there is more than one connected component in the graph due to physical constraints or other reasons, the number of sensors allocated to each connected component can be determined first, after which we run the map division algorithm for each component using the determined number of sensors. In the algorithm, we build a graph of the topology by treating the PoIs as vertices. We connect two PoIs with an edge if a line (of non-negligible width) connecting the centers of the PoIs does not pass through any other PoIs in the topology. This connection method ensures that if two PoIs are neighbors in the graph, a sensor can move between them without passing through any other PoIs. We then build each subregion sequentially by picking a PoI that satisfies the following three conditions: (i) it is a neighbor of any PoIs already included in the subregion, (ii) it is not a *cut vertex*¹, and (iii) after taking the vertex, either the subregion has the target threat level, or there exists a *non-cut vertex* to choose from to expand the subregion. Not picking any cut vertex in constructing a subregion is a sufficient but not necessary condition to ensure that the remaining subregions can be built to

¹A vertex is a cut vertex if its removal causes an increment in the number of connected components in the graph.

achieve the target threat level without overlapping with other subregions. For example, if we pick a cut vertex as part of a subregion, it is possible that the last subregion needs to include vertices from the two disconnected components formed after removing the cut vertex. In this case, the connection would require passing through other subregions in the network.

By following the above map division algorithm, we can guarantee that (i) PoIs of each subregion are *connected*, given that the original graph is connected, and (ii) subregions do not pass through each other. This is because we build a subregion by selecting only neighbors of PoIs that are already included in the subregion, i.e., all candidate PoIs must be connected to the subregion without passing through any other subregions. Moreover, the algorithm tries to achieve similar threat levels among all the subregions, by adding PoIs until a subregion's threat level reaches the target value. However, the goal is not guaranteed because as we grow a subregion, we may not be able to find a node that satisfies all the three selection criteria. When that happens, we have to stop growing the subregion even if the target threat level has not been reached.

- *Static division of sensor responsibility (with overlapping), MDO.* Experimental results show that although MD gives much improved performance in effective coverage, the matching metric could suffer seriously because subregions could be of very different threat levels. The key objective of the MDO algorithm is to divide the threat of the surveillance area *as evenly as possible* among the subregions. To do so, the MD algorithm is modified so that subregions may now overlap. This is achieved by modifying Line 12 in Fig. 2.2 to remove v from G only when all its threat is covered; otherwise, we subtract the covered threat from v and keep it at G . An example illustrating the output of the algorithm is shown in Fig. 2.3(b). Although the change from MD to MDO seems minor, experimental results show that MDO can out-perform MD significantly in terms of the matching metric. We will discuss in the following section how this change

```

1  Map_Division(topology  $t$ , number of sensors  $n$ )
    // graph building
2  for all non-zero threat cells  $i, j$ 
3      add  $edge_{i,j}$  to  $G$  if the edge does not pass through any other non-zero threat cell

    // the target threat level at each subregion
4   $target = threat(G)/n$ 

    // region partition
5  for  $i = 1$  to number of sensors  $-1$ 
6      let  $R_i$  be the set of cells in the  $i$ -th region
7       $R_i = \{\}$ 
8      find a vertex  $v$  in  $G$  such that at least one of its neighbors is not a cut vertex
9      remove  $v$  from  $G$  and add to  $R_i$ 

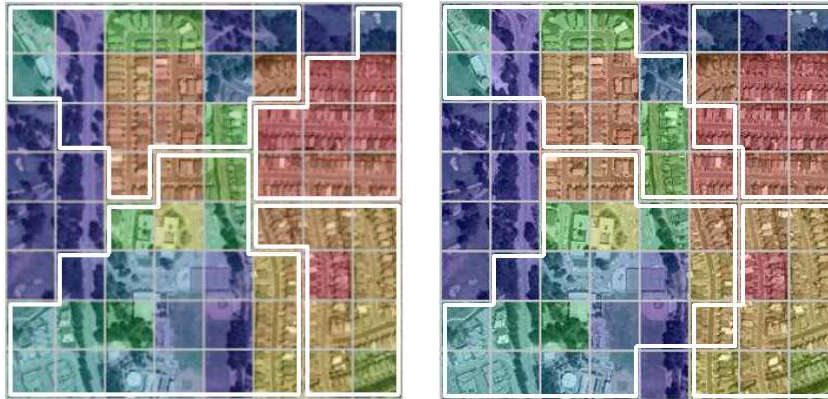
10     while  $threat(R_i) < target$ 
11         find a neighbor,  $v$ , from members of  $R_i$  such that
             $v$  is not a cut vertex and
            (there exists a vertex in  $v \cup R_i$  which has a non-cut vertex neighbor
            or  $threat(R_i) \geq target$  after adding  $v$ )
12         remove  $v$  from  $G$  and add to  $R_i$ 
13         if no such vertex exists, proceed to next subregion

14   $R_n = \{\text{all remaining vertices}\}$ 

15  output all  $R_i$ 

```

Figure 2.2. Specification of map division algorithm.



(a) Sub-regions for MD

(b) Sub-regions for MDO

Figure 2.3. Examples illustrating the output of the map division algorithms using the San Francisco city map for a 4-sensor case.

in map division helps improve the matching. Within the assigned subregion, each sensor follows the WRW-aLP algorithm. To avoid redundant coverage due to overlapping subregions, a sensor is not allowed to pick a cell that is being chosen as the destination by another sensor sharing the overlapping cell.

Analytical Study of WRW-aLP

Markov Model of WRW-aLP

We model the WRW-aLP algorithm by a Markov Chain for a single WRW-aLP sensor. Without loss of generality, we assume that the PoI cells of the surveillance region are denoted as cells $i = 1, \dots, n$ of the Markov model, and use cell 0 in the model to represent the collection of non-PoI cells in the surveillance region. The state of the Markov Chain is denoted by $X = (i, U)$, where i is the location of the sensor (i.e., the PoI the sensor is at), and U is the vector of undercoverage times as defined in Section 2.1.

Besides the notations in Section 2.1, it is convenient to introduce the following quantities. T_{ij} specifies the actual travel time of the sensor to travel from i to j . In addition, the numbers $T_{ij,k}$ specify how such a trip increases the coverage time of each

PoI k during the travel from i to j . We use $T_{ij,0}$ to denote the travel time overhead of the trip from i to j , i.e., the period during which the sensor is not within the range of any PoI. Hence, $\sum_{0 \leq k \leq n} T_{ij,k} = T_{ij}$. Finally, we use $D_{ij} = T_{ij} - T_{ij,0}$ to denote the duty time portion of the actual travel time. To admit more general models, we allow $i = j$, i.e., the sensor can decide to remain at its current location in a transition. In this case, T_{ii} is set to be zero.

Notice that all the quantities $\{T_{ij}\}$, $\{T_{ij,k}\}$ and $\{D_{ij}\}$ are deterministic properties of the surveillance region and the paths between pairs of PoIs. Thus, they can be pre-computed by the coverage algorithm.

A transition from state $X = (i, U^X)$ to state $Y = (j, U^Y)$ in the Markov Chain is labeled by (i, j) , a trip of the sensor from cell i to cell j . As the goal of the WRW-aLP algorithm is to reduce the undercoverage, the sensor tends to go to PoIs with large undercoverage. In the stochastic algorithm, the transition probability is chosen according to:

(1) Whether j is within a given distance from i . If so, we say that j is *eligible*. We indicate the condition by the indicator function $I(i, j)$, i.e., $I(i, j) = 1$ if j is within the allowed distance of i , otherwise $I(i, j) = 0$.

(2) The relative under-coverage times between the eligible PoIs. The probability that j is chosen as the next PoI is given by

$$Pr_{ij} = \frac{I(i, j) \times W_j U_j^+}{\sum_{1 \leq k \leq n} I(i, k) \times W_k U_k^+}, \quad (2.1)$$

where $U_k^+ = \max(U_k, 0)$ and W_k are the weights given as some positive numbers, e.g., $W_k = \Phi_k$.

In the degenerate cases that $I(i, k) = 0$ or $U_k^+ = 0$ for all k , we can simply choose the next PoI randomly, according to some uniform distribution. The precise choices of the W_k 's and what to do in the degenerate cases will not affect the qualitative results.

In WRW-aLP, once the sensor has moved from i to j , it will stay there for a pause time, $T_P(U_j)$, which can depend on j 's current undercoverage. The value of this pause

time is part of the decision of the transition. Hence, a movement from i to j increases the sensor's duty time by $D_{ij} + T_P(U_j)$. In addition, for each cell, its coverage time will be increased by $T_{ij,k}$ for $k \neq j$ and $T_P(U_k)$ for $k = j$. Using the above, the change of the undercoverage $U_k^Y - U_k^X$ (for $k = 1, \dots, n$) can be computed explicitly.

Note that the transition probability from X to Y given by (2.1) depends only on the current state i and U^X . Hence, it models a Markov Chain process.

Performance analysis: single sensor

This section analyzes the WRW-aLP Markov Chain model for the single sensor case. One of the main behaviors of any Markov Chain concerns its long time behavior and ergodicity and stationarity properties. It is natural to ask these questions to our WRW-aLP model. Many of the standard results are expected to be true. *However, due to the side effects of coverage—the sensor can pass by some unintended PoIs due to $T_{ij,k} > 0$ for $k \neq j$ —it is not immediately clear whether the long time behavior can be related to the desired threat-based coverage, or the threat profile can be achieved.*

For example, without the side effects, i.e., $T_{ij,k} = 0$, to achieve the threat profile Φ , we can simply construct a Markov Chain with Φ as its stationary distribution. Such a chain can be realized by appropriate Monte-Carlo simulations. However, with non-zero $T_{ij,k}$'s, the analysis needs to be modified with some care. Our main results state that the desired threat profile can *always* be achieved, provided that the pause time is *long enough* compared with the $T_{ij,k}$'s. They are proved through the concept of *Lyapounov function*, which appears in the study of the stability properties of dynamical systems.

To simplify the analysis, we set all the pause times $T_P(U_j)$'s equal to some fixed constant T_P . Furthermore, $I(i, j) = 1$ for all i and j , i.e., any PoI is accessible within one transition. Such an assumption can certainly be relaxed. In general, the pause time can even be random as in our WRW-aLP specification.

Consider the following quantity:

$$V(X) = \sum_{i=1}^n \alpha_i (U_i^X)^2, \quad (2.2)$$

where α_i 's are some weights to be determined. Clearly, $V = 0$ refers to perfect matching with the threat profile. Hence, the goal is, in the long run, to make V as *small* as possible. The following two results show that “on average,” V decreases as the mobile algorithm continues. To better illustrate the idea, we first consider the case of no side effects and treat the latter as a *perturbation*.

Theorem 2.3.1 *Let $T_{ij,k} = 0$ for $i, j, k = 1, \dots, n$. There exists a positive number $\lambda > 0$ such that if $V(X) \geq \lambda$, then*

$$E_X V(Y) \leq V(X) - 1, \quad (2.3)$$

where X and Y refer to the current and next states, respectively, and $E_X(\cdot)$ denotes the expectation given the state X . (The choice of the weights α_i 's will be specified in the proof of this Theorem.)

Proof Let the current and next sensor locations be i and j . Then the undercoverage of PoI k will be increased by $\Phi_k T_p$ for $k = j$ and $(\Phi_j - 1)T_p$ if $k = j$. Hence,

$$V(Y) = \sum_{k \neq j} \alpha_k (U_k^X + \Phi_k T_p)^2 + \alpha_j (U_j^X + (\Phi_j - 1)T_p)^2.$$

Then, the expectation $E_X(V(Y))$ equals

$$\sum_{j=1}^n \left[\sum_{k \neq j} \alpha_k (U_k^X + \Phi_k T_p)^2 + \alpha_j (U_j^X + (\Phi_j - 1)T_p)^2 \right] Pr_{ij}.$$

Now consider

$$\begin{aligned} & E_X V(Y) - V(X) \\ &= \sum_{j=1}^n \left[\sum_{k \neq j} \alpha_k (U_k + \Phi_k T_p)^2 + \alpha_j (U_j + (\Phi_j - 1)T_p)^2 - \sum_{k=1}^n \alpha_k (U_k)^2 \right] Pr_{ij} \\ & \quad (\text{note: } \sum_j Pr_{ij} = 1) \\ &= Z_*^{-1} T_p \left[2 \times I + II \right], \end{aligned}$$

where $Z_* = \sum_{k=1}^n W_k U_k^+$. The quantities I and II are defined and analyzed as follows.

$$\begin{aligned} I &= \sum_{j=1}^n \left(\sum_{k=1}^n \alpha_k \Phi_k U_k - \alpha_j U_j \right) W_j U_j^+ \\ &\leq \left(\sum_{k=1}^n \alpha_k \Phi_k U_k^+ \right) \left(\sum_{j=1}^n W_j U_j^+ \right) - \left(\sum_{k=1}^n \alpha_k W_k (U_k^+)^2 \right), \end{aligned}$$

where we have used $U \leq U^+$ and $UU^+ = (U^+)^2$. Using the Cauchy-Schwartz inequality, we have

$$\begin{aligned} \left(\sum_{k=1}^n \alpha_k \Phi_k U_k^+ \right) &\leq \left(\sum_{k=1}^n \Phi_k \right)^{1/2} \left(\sum_{k=1}^n \alpha_k^2 \Phi_k (U_k^+)^2 \right)^{1/2}, \\ \left(\sum_{k=1}^n W_k U_j^+ \right) &\leq \left(\sum_{k=1}^n \frac{W_k}{\alpha_k} \right)^{1/2} \left(\sum_{k=1}^n \alpha_k W_k (U_k^+)^2 \right)^{1/2}. \end{aligned}$$

By choosing $\Phi_k \alpha_k^2 = \alpha_k W_k$, i.e., $\alpha_k = W_k \Phi_k^{-1}$, then

$$\left(\sum_{k=1}^n \frac{W_k}{\alpha_k} \right)^{1/2} = \left(\sum_{k=1}^n \Phi_k \right)^{1/2} = 1,$$

which gives $I \leq 0$. On the other hand, equalities in the above Cauchy-Schwartz inequalities hold if and only if the vector $(U_k^+)_{k=1}^n$ is a multiple of $(\alpha_k \Phi_k)_{k=1}^n$ or $(W_k)_{k=1}^n$. This is impossible as $\sum_{k=1}^n U_k = 0$. We can thus infer the existence of a $\mu > 0$ such that

$$I \leq -\mu \sum_{k=1}^n \alpha_k (U_k^+)^2. \quad (2.4)$$

(See also Remark (B) on Page 27.)

Next, the quantity II equals:

$$II = T_P \sum_{j=1}^n \sum_{k=j}^n \alpha_k \Phi_k^2 + \alpha_j (\Phi_j - 1)^2 W_j U_j^+.$$

Note that it involves only *linear* terms of U_j^+ . Hence, there exists a $C > 0$ such that

$$II \leq C \sqrt{\sum_{k=1}^n (U_k^+)^2}.$$

So if $\sqrt{\sum_{k=1}^n (U_k^+)^2} \geq \lambda$ for some large number λ , we have

$$2 \times I + II \leq -\frac{\mu}{2} \sum_{k=1}^n \alpha_k (U_k^+)^2.$$

After taking into account of the prefactor Z_*^{-1} and re-defining the value of the constant μ , we get the desired result:

$$E_X V(Y) - V(X) \leq -\mu \sqrt{V(X)} < -1.$$

In the above, we have used the fact that for $\sum_k U_k = 0$, there exists a $C > 0$ such that

$$\sum (U_k^+)^2 \leq \sum (U_k)^2 \leq C \sum (U_k^+)^2.$$

■

The next result incorporates the presence of side effects, i.e., $T_{ij,k} > 0$. We use the same notation as in the previous Theorem.

Corollary 2.3.2 *There exist $\lambda, T_P^* > 0$ such that for $T_P > T_P^*$ and $V(X) > \lambda$, then*

$$E_X V(Y) \leq V(X) - 1.$$

Proof We will only outline the proof as it is very similar to Theorem 2.3.1. Due to the $T_{ij,k}$'s, the function $V(Y)$ evaluated at the new state is now given as:

$$\sum_{k=j} \alpha_k (U_k^X + \Phi_k T_P - T_{ij,k})^2 + \alpha_j (U_j^X + (\Phi_j - 1)T_P)^2.$$

Hence, the quantity $E_X V(Y) - V(X)$ equals

$$\sum_{j=1}^n \sum_{k=j} \alpha_k (U_k + \Phi_k T_P - T_{ij,k})^2 + \alpha_j (U_j + (\Phi_j - 1)T_P)^2 - \sum_{k=1}^n \alpha_k (U_k)^2 \quad Pr_{ij}.$$

We proceed as before. The quantity I now has an extra term given by:

$$- \sum_{j=1}^n \sum_{k=j} \alpha_k U_k \frac{T_{ij,k}}{T_P} \quad W_j U_j^+.$$

Note that if $\frac{T_{ij,k}}{T_P} \ll 1$, this extra term can be bounded by $c \sum_{k=1}^n (U_k^+)^2$ for some *small* constant c so that it can be absorbed by the $-\mu V(X)$ appearing in Theorem 2.3.1.

The quantity II becomes

$$T_P \sum_{j=1}^n \left[\sum_{k=j}^n \alpha_k \left(\Phi_k - \frac{T_{ij,k}}{T_P} \right)^2 + (\Phi_j - 1)^2 \right] W_j U_j^+.$$

Again, it can be considered in exactly the same way as before because only linear terms of U_k^+ 's are involved.

Combining the above statements about I and II , we have the desired conclusions.

■

[Remarks] (A) The above two results show that the positive quantity V decreases in expectation if it has a large value to start with. Hence on average, *large* values of V will be *reduced*. Combining with the theory of Martingales, we can show that

$$\lim_{T \rightarrow +\infty} \frac{1}{T} V(X(T)) = 0,$$

i.e., the overall undercoverage has at most *sub-linear* growth. This leads to the result that, *in the long run, the threat profile can be achieved exactly*.

(B) The number μ in Eqn (2.4) can be estimated. In fact,

$$\mu = 1 - \max_j \sqrt{\sum_{k=j} \Phi_k^2} (> 0).$$

Combining with the actual pre-computed values of the $T_{i,j,k}$'s, we can also estimate λ and most importantly, T_p^* . This can provide *practical* guidance of how long the pause time should be in order to ensure bounded global undercoverage.

Performance analysis: multiple sensors

The analysis of the algorithm in the multi-sensor case turns out to be quite intricate due to action of multiple factors including travel overheads, overlapping in the sensor trajectories, and redundancy in the destination occupancy by the sensors, among others. In realistic parameter settings, it is not easy to single out one particular factor and perform control experiments. Hence we will concentrate on extracting intuitions from the simulation results which already can demonstrate a broad range of possibilities. Our main message is that without proper sub-division of the network and sensor coordination, the overall performance can deteriorate easily.

Despite the complexity of exact analysis, the following theorem nevertheless gives better understanding of the matching performance of MD and MDO. Assume that

there are m sensors deployed, and we divide the network into m sub-regions, one for each sensor. We use Φ'_i and e_i to denote the threat level and effective coverage by the assigned sensor of sub-region i .

Theorem 2.3.3 *If the sensor assigned to a sub-region does not overlap with any other sensors at any time, perfect matching of the whole network can be achieved in the long run if and only if the condition*

$$\frac{e_0}{\Phi'_0} = \frac{e_1}{\Phi'_1} = \frac{e_2}{\Phi'_2} = \dots = \frac{e_m}{\Phi'_m} \quad (2.5)$$

holds.

Proof Since there is no overlap of the sensors, the matching performance of a sensor in each sub-region can be determined individually. In this case, the result of achieving the threat profile exactly in the long run for each sub-region follows directly from the analysis in the previous section.

The condition of Eq. (2.5) implies that sensors of different sub-regions spend the same amount of monitoring time per unit threat level. Since matching is achieved exactly in each sub-region in the long run, it follows that perfect matching of the whole area can also be achieved in the long run. Conversely, if perfect matching of the whole network is achieved in the long run, it implies that sensors spend the correct portion of time at each PoI. Hence, the monitoring time per unit threat level is the same among all of the sensors, and the condition of Eq. (2.5) holds. ■

Notice that the map division algorithm in MD or MDO does not explicitly consider the condition for perfect matching of the whole network. This is because it is hard to determine the effective coverage of a sensor in a subregion without actually deploying the sensor and measuring the performance. However, our experiments show that MDO out-performs MD significantly in terms of matching when the sensor density is high, which shows in turn that MDO is able to approximate the condition more closely than MD. The latter observation holds because in MDO, we divide the network into subregions of equal threat levels, i.e., $\Phi'_0 = \Phi'_1 = \dots = \Phi'_m$. Hence, the condition

for perfect matching in the whole network (Theorem 2.3.3) becomes $e_0 = e_1 = \dots = e_m$, which is more easily satisfied even without its explicit consideration in the map division. In particular, when the sub-regions contain no zero-threat cells, then we have $e_0 = e_1 = \dots = e_m = 1$ naturally.

2.3.2 Stateless — Markov chain

In the previous subsection we proposed and analyzed a stateful mobility algorithm. However, its performance in balancing matching and fairness is based on heuristics only, and there is no mathematical analysis to compute the optimal pause time to achieve the desired balance in terms of matching and fairness. In this subsection, we consider a stateless mobility algorithm in which the decision to pick the next destination is based on the current position and a transition matrix. The mobility algorithm is stateless because the decision for the next destination only depends on the current position, and the decision can be made in constant time by flipping a coin. Hence, the movement of the mobile node can be modeled using a Markov chain. We demonstrate how the cost of the mobility algorithm can be expressed in terms of the transition matrix. And we propose a steepest descent algorithm to compute the optimal parameters to minimize the cost function.

Problem Formulation

Scheduling of our mobile sensor using the stateless mobility algorithm is controlled by a Markov chain, which describes the sensor’s travel between the PoIs. We design the Markov chain to minimize a general cost function, which can account for factors such as the distributions of per-PoI coverage and exposure times, an entropy measure quantifying the randomness of the sensor schedule, information capture about dynamic events, energy efficiency, etc. A key feature of our framework is that it can incorporate possibly conflicting performance objectives while accounting for the dependencies between the PoIs covered due to their geographical placement.

For the sake of discussion, we will focus on a cost function, denoted as U , that considers the distributions of per-PoI coverage times and per-PoI exposure times. Other performance criteria can be included without much difficulty. Our approach expresses the cost function in terms of the transition probabilities of the Markov chain and uses steepest descent to find the optimal transition probabilities. The advantages of our method are its *simplicity* and the *comprehensiveness of the search space*.

Derivation of the cost function

We now describe details of the mathematical set-up. Suppose there are M PoIs, denoted by the set $S = \{1, 2, \dots, M\}$. Consider a *time-homogeneous* Markov chain $\{X_n\}_{n \geq 1}$ taking values in S . The sensor moves between the PoI locations (i.e., states in the Markov chain) as the Markov chain state transitions. The transition probabilities are denoted by $\{p_{ij}\}_{i,j \in S}$: $p_{ij} = P(X_{n+1} = j | X_n = i)$.

In order to relate the (discrete) transition time step n and the physical elapsed time, we use the following notation:

- T'_{jk} equals the *sum* of the *travel time* from PoI j to PoI k and the *pause time* P_k at k . Note that $T'_{jj} = P_j$.
- $T'_{jk,i}$ equals the time the sensor, upon traveling from j to k , passes by PoI i . These quantities in essence reflect the side effects of coverage imposed by geographical constraints: A PoI, say i , can still be covered even if it is not the intended destination of a transition, because the sensor has to go through i in order to reach the destination. We use the following convention: $T'_{jj,i} = 0$ for $j = i$ and $T'_{jj,j} = P_j$ (the pause time at j).

With the above, the physical time elapsed after N state transitions is given by:

$$T(N) = \sum_{n=1}^N \sum_{j,k} T'_{jk} \delta_j(X_n) \delta_k(X_{n+1}),$$

where $\delta_i(x) = 1$ if $x = i$ and 0 otherwise. Furthermore, the total coverage time of PoI i by the sensor is given by:

$$C_i(N) = \sum_{n=1}^N \sum_{j,k} T'_{jk,i} \delta_j(X_n) \delta_k(X_{n+1}).$$

Next we define $\langle E_i(N) \rangle$, the average *exposure time*, to be the *arithmetic average* (taken in the first N transitions) of the lengths of the *continuous* time intervals during which the PoI i is *out of range* of the sensor. More precisely, let $E_{i,1}, E_{i,2}, \dots, E_{i,m}$ be the consecutive time intervals during which the sensor is away from i . Then

$$\langle E_i(N) \rangle = \frac{E_{i,1} + E_{i,2} + \dots + E_{i,m}}{m}.$$

With the above, we define the cost function U to be the following long time limit:

$$\lim_{N \rightarrow \infty} \left\{ \sum_{i \in S} \frac{1}{2} \alpha_i \left(\frac{C_i(N)}{T(N)} - \Phi_i \right)^2 + \sum_{i \in S} \frac{1}{2} \beta_i \left(\langle E_i(N) \rangle \right)^2 \right\}, \quad (2.6)$$

where the α_i 's and β_i 's are user-defined constants, and Φ_i 's represent the prescribed distribution of per-PoI coverage times.

The intuition of the above cost function is as follows. The part with the α_i terms measures the discrepancy between the actual and prescribed distributions of the per-PoI coverage times. The part with the β_i terms measures the average per-PoI exposure time. Intuitively, minimizing the function U attempts to reduce both the coverage-time discrepancy and expected exposure time. However, the coverage-time and exposure-time measures are generally antagonistic so that the optimal choice of the transition probabilities is the result of a delicate trade-off between them. The parameters α_i 's and β_i 's reflect the relative importance between the two measures and they can be adjusted by the user.

Our next goal is to express the above cost function as an explicit function of the p_{ij} 's which completely characterize the Markov chain. For this, we will assume for the rest of the paper that the Markov chain is ergodic.

Let $\{\pi_i\}_{i \in S}$ be the stationary distribution of the Markov chain. By ergodicity, we have:

$$\lim_{N \rightarrow \infty} \frac{T(N)}{N} = \sum_{j,k} \pi_j p_{jk} T'_{jk} \quad \text{and} \quad \lim_{N \rightarrow \infty} \frac{C_i(N)}{N} = \sum_{j,k} \pi_j p_{jk} T'_{jk,i}.$$

Hence the long term average of the coverage time distribution \bar{C}_i can be defined and computed as:

$$\bar{C}_i = \lim_{N \rightarrow \infty} \frac{C_i(N)}{T(N)} = \frac{\sum_{j,k} \pi_j p_{jk} T'_{jk,i}}{\sum_{j,k} \pi_j p_{jk} T'_{jk}}. \quad (2.7)$$

In order to simplify the mathematical expression for the long time limit $\lim_{N \rightarrow \infty} \langle E_i(N) \rangle$, we will make the following assumptions which will not change our conclusions qualitatively:

- If a PoI is simply being passed by while the sensor is traveling between two other PoIs in a transition, then the passing-by is not considered a return visit.
- We assume that the time elapsed during each travel is always equal to one unit of time — this assumption is only enforced in the computation of the $\langle E_i \rangle$'s.

With the above simplifications, physically each exposure time segment $E_{i,j}$ is measured from the PoI location (e.g., j) immediately after the sensor has *left* i . Then necessarily, $j = i$. Let R_{ji} be the *expected value* of the time the sensor takes to reach (or return to) i from j . This quantity is also traditionally called the *expected first passage time*. As the probability of traveling from i to j is given by p_{ij} , we have the following expression for the long time average of the exposure times:

$$\bar{E}_i = \lim_{N \rightarrow \infty} \langle E_i(N) \rangle = \frac{\sum_{j=i} p_{ij} R_{ji}}{\sum_{j=i} p_{ij}} = \frac{\sum_{j=i} p_{ij} R_{ji}}{1 - p_{ii}}. \quad (2.8)$$

With the above, the cost function U can be written as:

$$U = \sum_{i \in S} \frac{1}{2} \alpha_i \sum_{j,k} \pi_j p_{jk} (T'_{jk,i} - \Phi_i T'_{jk})^2 + \sum_{i \in S} \frac{1}{2} \beta_i \left[\frac{\sum_{j=i} p_{ij} R_{ji}}{1 - p_{ii}} \right]^2. \quad (2.9)$$

It is now time to remark about the search space of our optimization. In the current setting, we search within the space of *all* transition probabilities. This is in contrast to most problems in stochastic control, in which the search or *action space*

is usually parameterized by a finite number of parameters. Thus the novelty of our approach is that the solution gives the *true* optimal among the class of *all* Markov chains. However, the solution space also becomes significantly more complex than less comprehensive formulations, and we will solve the problem by properly adding noise in the optimization procedure.

Analytical representation of U using generalized inverse

Even though the formula (2.9) for the cost function is highly explicit, it still requires the computation of the stationary distribution $\{\pi_i\}$ and the first passage time matrix $\{R_{ij}\}$. In order to compute them efficiently, we employ the concept of *generalized inverse*. In the following, we review the basic properties of this mathematical term, with particular application towards Markov transition matrices. The material is discussed in [28].

Let $P = \{p_{ij}\}_{i,j \in S}$ be an ergodic Markov transition matrix. Consider the generalized inverse A^\sharp of $A = I - P$ which is defined as the matrix satisfying:

$$AA^\sharp A = A, \quad A^\sharp AA^\sharp = A^\sharp, \quad \text{and} \quad A^\sharp A = AA^\sharp.$$

The existence and computation of A^\sharp is given by [28, Thm. 2.1, 5.1].

Using A^\sharp , the stationary distribution $\pi = \{\pi_i\}$ and the expected first passage time matrix $R = \{R_{ij}\}$ can be expressed as:

$$W = I - AA^\sharp \quad ([28, \text{Thm. 2.3}], \tag{2.10}$$

$$R = (I - A^\sharp + JA_{dg}^\sharp)D \quad ([28, \text{Thm. 3.3, 3.6}], \tag{2.11}$$

where W is the matrix such that all of its rows are the stationary distribution π ; J is the $M \times M$ square matrix with all of its entries equal to one; A_{dg}^\sharp is the diagonal matrix consisting of the diagonal entries of A^\sharp ; and D is the diagonal matrix such that $D_{ii} = \frac{1}{\pi_i}$. Note that we can have such a simple expression (2.11) for the R_{ij} 's because of the simplifying assumptions we have for the exposure times.

The generalized inverse is also related to the commonly known *fundamental matrix* $Z = (I - P + W)^{-1}$ by the formula [28, Thm. 3.1]:

$$Z = I + PA^\sharp. \quad (2.12)$$

Using Z , R can then be expressed as

$$R = (I - Z + JZ_{dg})D, \text{ or component-wise, } R_{ij} = \frac{\delta_{ij} - z_{ij} + z_{jj}}{\pi_j}. \quad (2.13)$$

The above result is the one used in the actual numerical computation.

Note that in principle the p_{ij} 's satisfy the constraints $0 \leq p_{ij} \leq 1$ and $\sum_{j \in S} p_{ij} = 1$, i.e., they lie in a higher dimensional polytope. Even though this property is manageable, to simplify the computational algorithm, we add a small penalization term to handle the case of $p_{ij} = 0$ or 1. More specifically, we consider the following penalized version U_ϵ of U :

$$\begin{aligned} U_\epsilon = & \sum_i \frac{1}{2} \alpha_i \sum_{j,k} \pi_j p_{jk} (T'_{jk,i} - \Phi_i T'_{jk})^2 \\ & + \sum_i \frac{1}{2} \frac{\beta_i}{\pi_i^2 (1 - p_{ii})^2} \sum_{j=i} p_{ij} (\delta_{ji} - z_{ji} + z_{ii})^2 \\ & - \sum_{ij} \frac{1}{\epsilon} \left[\ln p_{ij} \right] (\epsilon - p_{ij})^2 \text{sgn}(\epsilon - p_{ij}) \\ & - \sum_{ij} \frac{1}{\epsilon} \left[\ln(1 - p_{ij}) \right] (1 - \epsilon - p_{ij})^2 \text{sgn}(p_{ij} - 1 + \epsilon), \end{aligned} \quad (2.14)$$

where $\text{sgn}(x)$ equals one for $x \geq 0$ and zero otherwise. Note that $U_\epsilon \rightarrow +\infty$ as any of the p_{ij} converges to 0 or 1. This is prohibited by the steepest descent algorithm whose purpose is to minimize the U . The constraint $\sum_j p_{ij} = 1$ can be handled easily by a projection method to be described later.

Steepest Descent Optimization

This section describes the use of *steepest descent* to search for the minimum of U_ϵ . Let $U = U(q_1, q_2 \dots q_l)$ be some function depending on the variables $q_1 \dots q_l$. We

would like to search for the minimum of U by evolving the variable q_i 's. For simplicity, let $Q(t) = (q_1(t), \dots, q_l(t))$ denote the values of the q_i 's at time t . Then, we compute:

$$\frac{d}{dt}U(Q(t)) = \nabla U(Q(t)) \frac{dQ(t)}{dt},$$

Now if we choose

$$\frac{dQ(t)}{dt} = -\nabla U(Q(t)),$$

then

$$\frac{d}{dt}U(Q(t)) = -|\nabla U(Q(t))|^2 < 0,$$

i.e., U is *decreasing*! Note that steepest descent as implemented above could only lead to a *local minimum* or *critical point*. Later in this section we will show how *stochastic perturbation* can solve this problem.

To apply the above formalism for the minimization of the cost function, we write U as:

$$U = U(\pi, Z, P) = U(\pi(P), Z(P), P) = U(P),$$

where $P = (p_{ij})$ is the transition matrix. Then

$$\frac{d}{dt}U(P(t)) = \sum_i \frac{\partial U}{\partial \pi_i} \frac{d\pi_i(t)}{dt} + \sum_{jk} \frac{\partial U}{\partial z_{jk}} \frac{dz_{jk}(t)}{dt} + \sum_{jk} \frac{\partial U}{\partial p_{jk}} \frac{dp_{jk}(t)}{dt}.$$

The expressions for $\frac{\partial U}{\partial \pi_i}$, $\frac{\partial U}{\partial z_{ij}}$ and $\frac{\partial U}{\partial p_{ij}}$ can be easily derived by straightforward partial differentiation. The formula for $\frac{d\pi_i(t)}{dt}$ and $\frac{dz_{jk}(t)}{dt}$ can be obtained by considering the *perturbation analysis* of Markov chain. For this, we follow [29] closely.

- By [29, Formula (23c), (15), and (17)]:

$$\frac{d\pi}{dt} = \pi \dot{P} Z, \text{ or component-wise, } \frac{d\pi_i}{dt} = \sum_{k,j} \pi_k z_{ji} \dot{P}_{kj}.$$

- By [29, Formula (23d), (16), and (18)]:

$$\frac{dZ}{dt} = Z \dot{P} Z - W \dot{P} (Z^2), \text{ or component-wise, } \frac{dz_{ij}}{dt} = \sum_{kl} [z_{ik} z_{lj} - \pi_k (Z^2)_{lj}] \dot{P}_{kl},$$

where $(Z^2)_{lj} = \sum_m z_{lm} z_{mj}$.

Using the above, we have $\frac{dU}{dt}$ equals

$$\begin{aligned}
& \sum_i \frac{\partial U}{\partial \pi_i} \frac{d\pi_i}{dt} + \sum_{ij} \frac{\partial U}{\partial z_{ij}} \frac{dz_{ij}}{dt} + \sum_{ij} \frac{\partial U}{\partial p_{ij}} \frac{dp_{ij}}{dt} \\
&= \sum_i \frac{\partial U}{\partial \pi_i} \sum_{k,l} \pi_k z_{li} \dot{P}_{kl} + \sum_{ij} \frac{\partial U}{\partial z_{ij}} \sum_{kl} [z_{ik} z_{lj} - \pi_k (Z^2)_{lj}] \dot{P}_{kl} + \sum_{ij} \frac{\partial U}{\partial p_{ij}} \frac{dp_{ij}}{dt} \\
&= \sum_{kl} \sum_i \pi_k z_{li} \frac{\partial U}{\partial \pi_i} + \sum_{ij} \frac{\partial U}{\partial z_{ij}} [z_{ik} z_{lj} - \pi_k (Z^2)_{lj}] + \frac{\partial U}{\partial p_{kl}} \dot{P}_{kl}.
\end{aligned}$$

If we let $[D_P U]_{kl}$ be the expression

$$\sum_i \pi_k z_{li} \frac{\partial U}{\partial \pi_i} + \sum_{ij} \frac{\partial U}{\partial z_{ij}} [z_{ik} z_{lj} - \pi_k (Z^2)_{lj}] + \frac{\partial U}{\partial p_{kl}}, \quad (2.15)$$

then we have:

$$\frac{dU}{dt} = \sum_{kl} [D_P U]_{kl} \dot{P}_{kl} = \langle D_P U, \dot{P} \rangle.$$

We can simply take: $\dot{P}_{kl} = -[D_P U]_{kl}$. However, as mentioned earlier, $P(t)$ must be a transition matrix at all t , i.e., it has to satisfy the constraint that for all j , $\sum_k p_{jk}(t) = 1$. To accommodate this, we *project orthogonally* $D_P U$ onto this linear subspace, i.e.,

$$\frac{dU}{dt} = \langle \Pi[D_P U], \dot{P} \rangle,$$

and then take:

$$\dot{P}_{kl} = -(\Pi[D_P U])_{kl}.$$

The formula for the projection Π is given by:

$$\Pi_{ij} = U_{ij} - \frac{\sum_k U_{ik}}{M}, \quad \text{for all } i, j, \quad (2.16)$$

where U_{ij} and Π_{ij} are the entries of $[D_P U]$ and $\Pi[D_P U]$, respectively. Note that $\sum_j \Pi_{ij} = 0$, i.e., the sum of each row of Π is *zero*. Hence, the property that the quantities represent a transition probability matrix is preserved at all time.

Computational Algorithm

The above described steepest descent algorithm will be implemented as follows:

1. Start with an arbitrary ergodic transition probability P .
2. Compute $[D_P U]$ (2.15) and its projection $\Pi[D_P U]$ (2.16).
3. Set $V = -\Pi[D_P U]$.
4. Set the new P as $P + V * \Delta t$, where Δt is some small time step.
5. For the new P , compute π , Z and R by (2.10), (2.12) and (2.13).
6. Go back to step two, or stop if the optimal is attained (within some tolerance level).

Note that the ergodicity of P is ensured by the finiteness of all the R_{ij} 's which is indeed preserved during the whole computation.

In the rest of the chapter, we will study the following variants of the steepest descent algorithm:

V1: Basic algorithm. All the p_{ij} values are initially set to $\frac{1}{M}$, where M is the number of PoIs. A constant time step Δt is used. This provides a basic test for the validity of our steepest descent algorithm.

V2: Random initial data. In this case, the initial values of the p_{ij} 's are random while satisfying the constraints that $\sum_j p_{ij} = 1$ and $0 \leq p_{ij} \leq 1$. This test will ensure that our algorithm is stable with respect to the initial condition.

V3: Adaptive time step Δt . With the gradient information ∇U in the current transition matrix P , the optimal time step Δt_* is chosen as follows:

$$\Delta t_* = \min_{\delta > 0} U(P - \delta \nabla U(P)).$$

The boundaries of δ are first determined with respect to the constraint that $0 \leq p_{ij} \leq 1$ after the update. As it is unclear how the utility changes within the range of δ , a conservative *trisection method* (wherein only one sub-section is removed as we tighten the bounds) is used to tighten the range until Δt_* is found. The algorithm terminates when $\Delta t_* = 0$. We define the terminating point as a *local optimum* when in fact there exists another terminating point with a lower cost.

V4: Stochastic perturbations. This step is to make sure that our algorithm will not get stuck at a local minimum of the cost function, which is shown to be essential in the next section. The $[D_P U]$ values are perturbed by mean zero Gaussian noise with standard deviation σ . Δt_* is then computed using the perturbed $[D_P U]$. If $\Delta t_* = 0$, then $\Delta t = rand$, where *rand* is a random number between the boundaries of Δt , which are determined with respect to the constraint that $0 \leq p_{ij} \leq 1$ after the update. The updates to p_{ij} 's are accepted if the computed utility is better than the original values; otherwise, they are accepted with probability $e^{\frac{-\Delta_U}{k \times \log(count)}}$, where Δ_U is the worsening in cost U after normalization by the *best cost* computed so far, *count* is the number of iterations already performed by the algorithm, and k is a constant parameter. The normalization is important because it allows us to determine how high the algorithm should “jump” (in trying to get out of a local optimum) without knowing the range of the cost function U_ϵ (which is frequently unknown beforehand). Intuitively, the algorithm accepts unattractive changes with higher probability at the beginning of the search, and this probability decreases as the algorithm proceeds. This way of accepting the updates is similar to the cooling process in simulated annealing. The optimality and convergence of simulated annealing has a strong theoretical basis [30]. However, the proofs are mostly of theoretical interest, since the convergence in practice is usually much faster, as we show in Section 2.4.3.

Henceforth in this chapter, we will use *steepest descent algorithm* to refer to our general solution approach. Additionally, we will call variant V1 of the algorithm the *basic algorithm*, the basic algorithm modified by variants V2 and V3 the *adaptive algorithm*, and the basic algorithm modified by all of variants V2–V4 the *stochastically perturbed algorithm* or simply *perturbed algorithm*. We have verified the performance of the steepest descent by using the transition matrices computed by the algorithm at each iteration to drive a corresponding simulation of the mobile sensor’s coverage. We found that the algorithm is able to achieve good tradeoffs between the coverage-time and exposure-time metrics, as the weightings of the two parameters in the cost function are varied. Also, the adaptive algorithm can speed up convergence to the

final solution, and the perturbed algorithm can additionally converge to the globally optimal steady state in practically all of the scenarios.

2.4 Evaluation

Here we evaluate the surveillance properties of the mobility algorithms quantified using the following three metrics,

- *Matching between the actual coverage time and the targeted coverage time of a cell.* This metric quantifies the degree the mobility algorithm achieves the desired allocation of resource according to the threat-level of the cells. The closer the matching, the better the performance because the mobility algorithm gives a better approximation for the threat-based protection.
- *Unfairness.* This metric quantifies the weighted average amount of time a cell is exposed to threats without being monitored by any of the mobile sensors. A smaller value of unfairness is better as it means events happening at a cell can be captured in a shorter time by the mobile sensors.
- *Effective coverage.* This metric quantifies the fraction of time the sensors are monitoring effectively. As we assume that the sensing range of the mobile sensors can be approximated with a fixed value, and using more than one sensor to monitor a cell does not improve the accuracy of the reported value, a sensor is not monitoring effectively when it travels on cells of zero threat level, or it covers a cell that is already being monitored by another sensor.

2.4.1 Performance of the WRW Mobility Algorithm

Here we first present the performance of variants of the WRW Mobility Algorithm. We then study how the pause time of the WRW-aLP algorithm can be adjusted to achieve a balance between *matching* and *fairness*.

Performance of Variants of the WRW Mobility Algorithm

We evaluate the performance of the WRW-aLP algorithm in Section 2.3.1. We show the results for a real-life topology, namely a residential region in San Francisco. The map of the region is shown in Fig. 2.4(a). The region is of size 2000 feet by 2000 feet. It is divided into 8×8 cells, 51 of them are PoIs. The threat level of a cell is taken to be the number of residents in that cell, estimated from the LandScanTM 2004 database of population data. Figs. 2.4(c)–(f) show that progressively adding the a, L, and P features to WRW achieves actual coverage that increasingly match the threat profile in Fig. 2.4(b). WRW-aLP achieves the threat profile exactly when $P = 30$ minutes (Fig. 2.4(f)). This important property of exact threat-based coverage is further investigated in Fig. 2.5 for WRW-aLP. The figure shows a log-log plot of the growth rate of the global undercoverage time (as a percentage of the sensor duty time) against the pause time parameter P , for San Francisco and also two other cities Chicago and San Jose. Note that as P increases, the growth rate decreases. The log-log scale of the plot amplifies the effect of *zero* rate of growth as P becomes larger than some *critical value*.

Performance of WRW-aLP – Adjusting the Pause Time Parameter

We illustrate the impact of the pause time parameter P on the matching and unfairness measures, for the case of one mobile sensor using the WRW-aLP algorithm. In this set of experiments, we vary the pause time parameter to be $P = 1, 2, 4, 8, 16, 32,$ and 64 time units. Fig. 2.7 show combined plots of the RMSE (left y-axis) and the unfairness (right y-axis) as a function of P , for Atlanta and LA. The city map and the corresponding threat profile of the two cities are shown in Fig. 2.6. Notice that for both figures, as the pause time increases, (1) the unfairness increases, in a partly constant, partly linear manner; and (2) the RMSE decreases like $1/(P + c)$, where c is a small constant. We also show the 25- and 75-percentiles of the RMSE in Table 2.1 for the set of runs for Atlanta. Notice that the values deviate little from

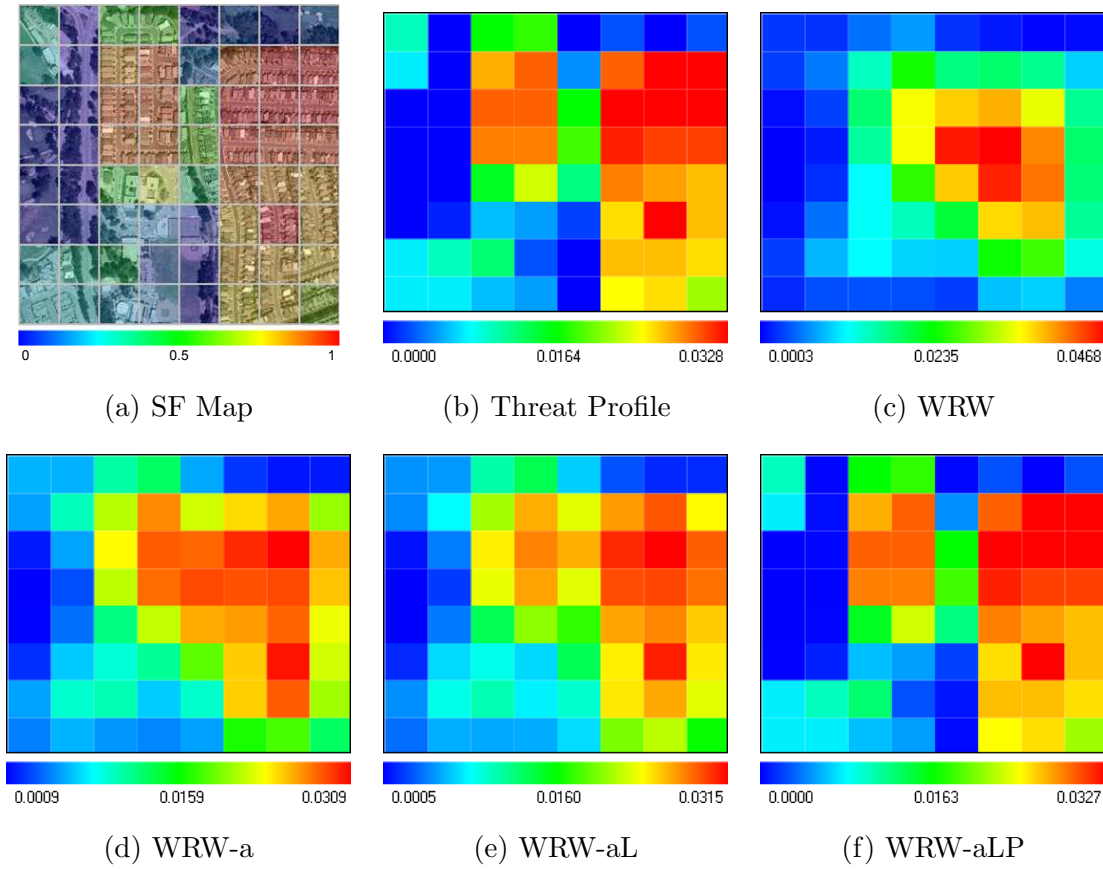


Figure 2.4. (a) Map of residential area in San Francisco. (b) Threat profile of the residential area. (c)–(f) Actual coverage achieved by progressive variants of WRW.

the averages. We will omit the 25- and 75-percentiles of the data distributions for the future sets of experiments, due to their closeness to the means. From this set of experiments, we conclude that there is an inherent tradeoff between the matching and fairness of coverage, and that the pause time parameter provides a means to control this tradeoff for the WRW-aLP algorithm.

Table 2.2 lists the fraction of time a sensor moves under WRW-aLP for different maximum sensor speeds and pause time in San Francisco (Threat profile is shown in Fig. 2.4(a).) It verifies that the amount of sensor movement required by WRW-aLP is practically small.

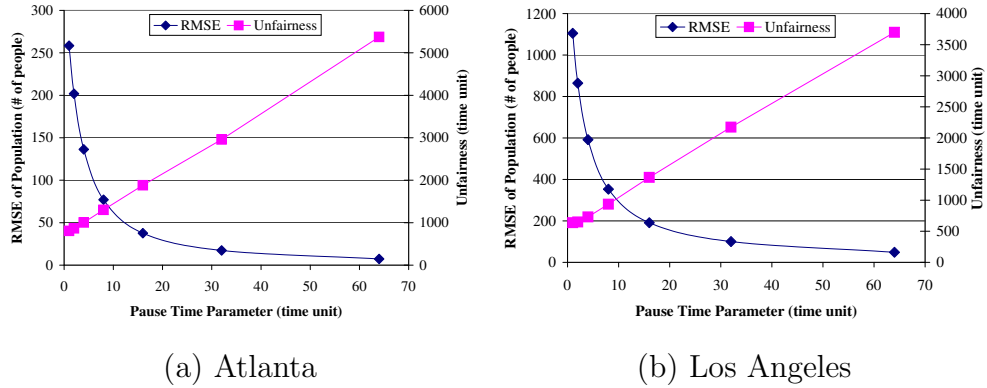
Figure 2.7. RMSE/unfairness tradeoff by P of WRW-aLP.

Table 2.1
Average and 25-/75-percentiles of WRW-aLP RMSE of population distribution for Atlanta, as a function of P .

P (time unit)	1	2	4	8	16	32	64
RMSE average	258.44	201.83	136.35	76.98	37.69	17.3	7.32
25-percentile	258.32	201.57	136.12	76.61	37.6	17.28	7.28
75-percentile	258.76	201.89	136.59	77.21	37.75	17.34	7.35

Table 2.2
Fraction of time sensor moves under WRW-aLP, for different maximum sensor speeds v and pause time parameters P .

P (min)	Maximum sensor speed v (mph)						
	0.4	0.6	0.8	1.2	2.4	3.6	4.8
1.3	0.983	0.975	0.967	0.950	0.900	0.852	0.807
2.7	0.967	0.950	0.933	0.900	0.807	0.725	0.658
13	0.837	0.765	0.702	0.601	0.422	0.323	0.260

2.4.2 Performance of Different Coordination Approaches among Multiple Sensors

Here we present the performance of different coordination approaches when more than one sensor is deployed in the network. As we have shown in the previous section

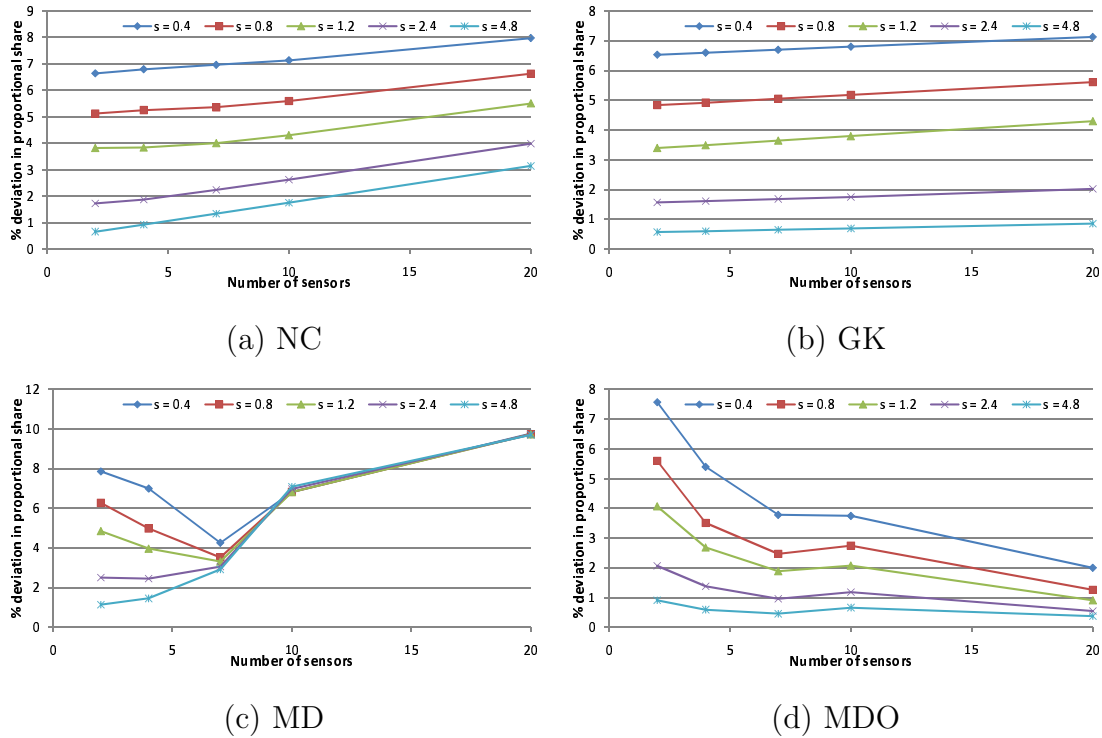


Figure 2.8. Percentage deviation of WRW-aLP global coverage profile from given threat profile, for multiple sensors under NC, GK, MD, and MDO in San Francisco.

that WRW-aLP gives the best balance between matching and fairness, we assume that in this section, no matter how sensors coordinate with each other, they all follow the WRW-aLP mobility algorithm.

This set of experiments illustrates the performance of NC, GK, MD, and MDO coordination (Section 2.3.1) under different sensor densities. Figs. 2.8, 2.9, 2.10, and 2.11 show the percentage deviation from the threat profile of WRW-aLP, unfairness, effective coverage, and pause fraction of the algorithm for San Francisco, respectively. The number of sensors is varied to be 2, 4, 7, 10 and 20. Notice from the figures that the trends for the performance of MD and MDO may not be smooth when we increase the number of sensors. It is because as we increase the number of sensors, we are also changing the way the area is divided among the sensors, and the performance of the algorithm depends highly on the resulting division.

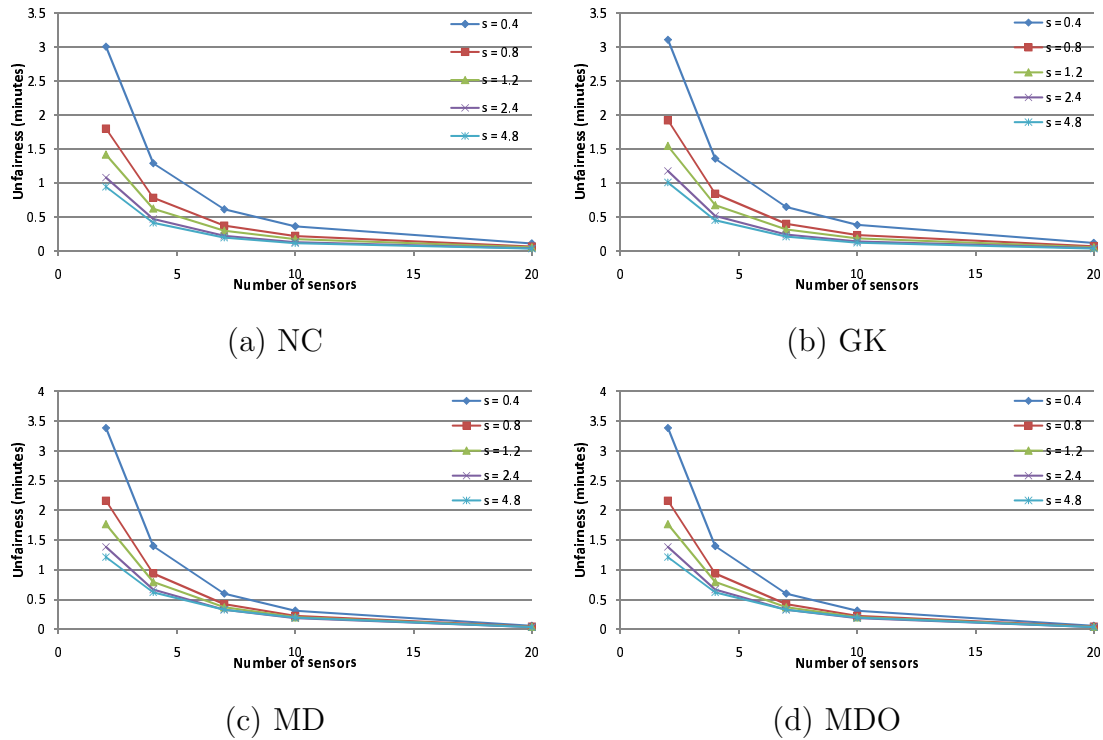


Figure 2.9. Unfairness of WRW-aLP global coverage profile from given threat profile, for multiple sensors under NC, GK, MD, and MDO in San Francisco.

Fig. 2.11 shows that for all coordination approaches, the sensors move for a small fraction of the time only, verifying the *limited mobility* property of WRW-aLP. Notice from Fig. 2.9 that for all coordination approaches, the unfairness is roughly reduced by half when we double the number of sensors, showing that sensor coordination will likely not further improve the unfairness beyond independent deployment of multiple sensors. Figs. 2.8(a) and 2.8(b) show that in contrast to unfairness, the steady-state matching does *not* improve as we use more sensors for NC and GK. This is because having more sensors will exacerbate the coverage redundancy, thus hurting the matching metric. However, for the high sensor speed shown in Fig. 2.8, GK achieves a 60% improvement over NC in terms of the percentage deviation (of the global coverage profile) from the threat profile when the sensor density is high. It is because a sensor

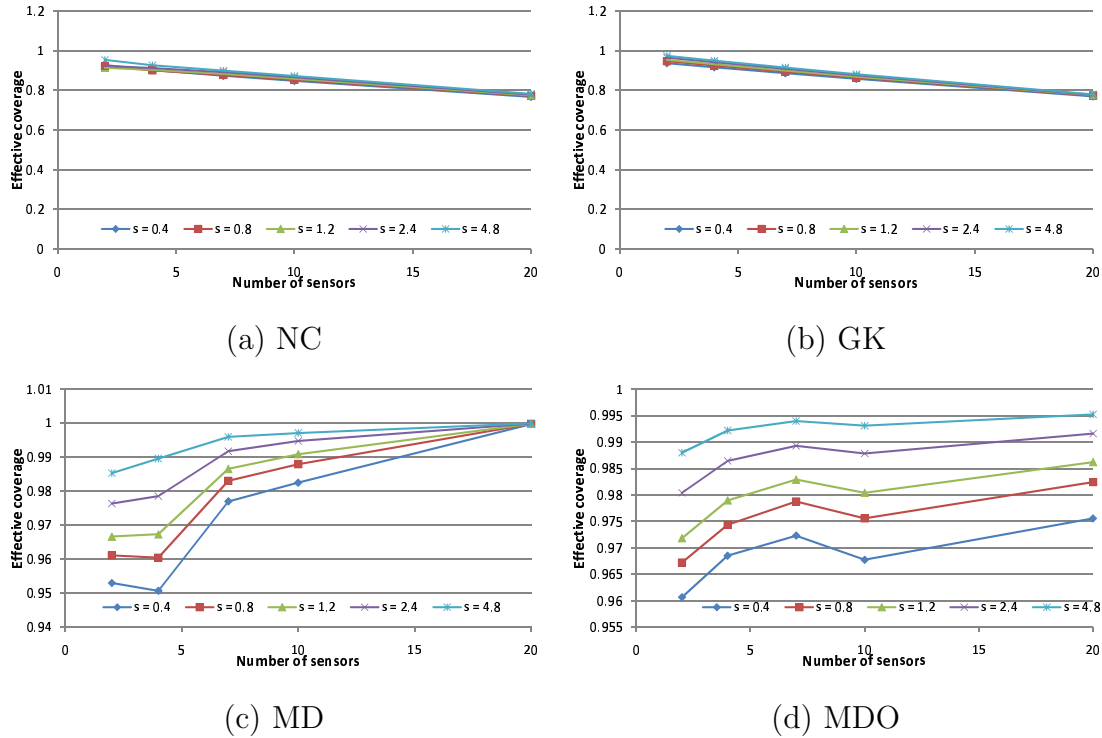


Figure 2.10. Effective coverage of WRW-aLP global coverage profile from given threat profile, for multiple sensors under NC, GK, MD, and MDO in San Francisco.

under GK can exploit the actual global coverage to better compensate for matching inaccuracies occurring from prior redundant coverage.

MD is an explicit attempt to solve the coverage redundancy problem and reduce the travel overhead. Fig. 2.10 shows that as the sensor speed is increased under higher sensor density, MD can achieve an effective coverage of more than 95%. Matching, however, may be significantly worsened when there are more sensors. This is because as the sensor density increases, the size of a subregion assigned to a sensor becomes smaller. It then becomes more difficult to divide the surveillance area into subregions of adjacent cells having similar threat levels. The unfairness achieved by MD also gets worsened with higher sensor densities, as shown in Fig. 2.9. It is because under MD, the number of possible destination cells of a sensor is much reduced with more sensors. Hence, it is more likely for the sensor to stay at the current PoI. This results

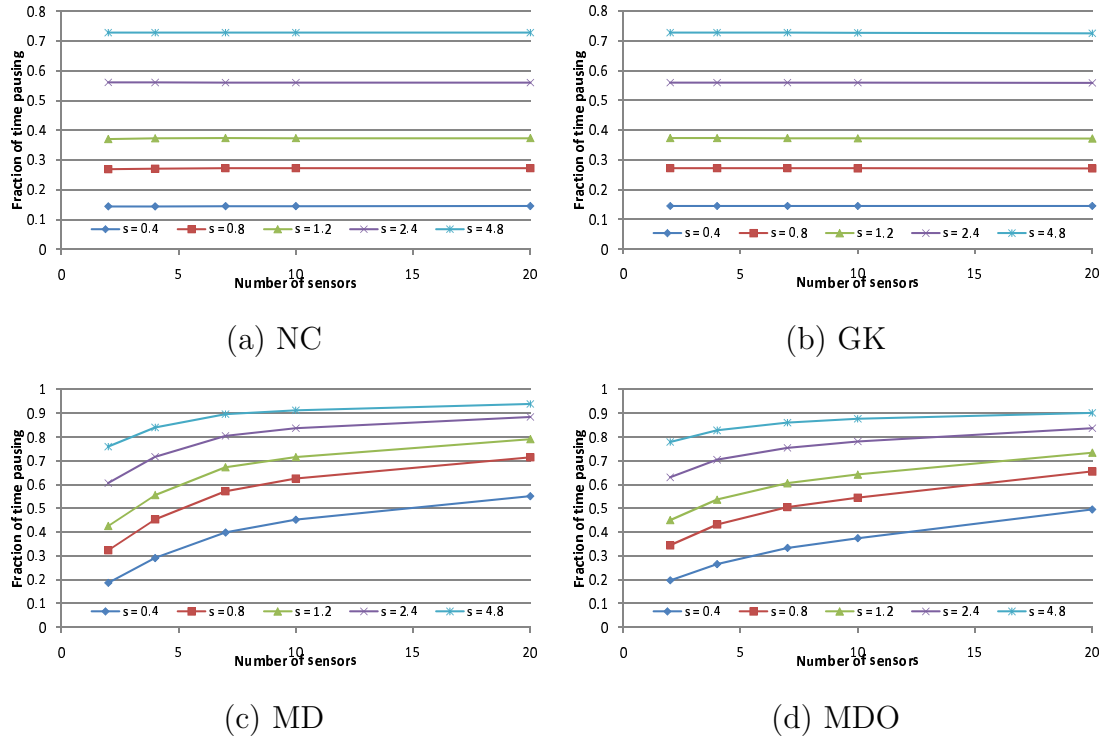


Figure 2.11. Pause fraction of WRW-aLP global coverage profile from given threat profile, for multiple sensors under NC, GK, MD, and MDO in San Francisco.

in higher unfairness as the sensor travels less and has longer pause times as shown in Fig. 2.11.

MDO is an attempt to solve the matching problem experienced by MD while enjoying its performance in effective coverage. We can observe from the figures that this approach is successful in achieving a good effective coverage while fairness is only slightly worsened. When the total number of mobile sensors is large, the matching performance of MDO can even outperform NC and GK because MDO has no mismatch caused by redundant coverage.

In summary, NC and GK give better matching than MD when the sensor speed or the sensor density is high, but they do so at the cost of significantly reduced effective coverage. On the other hand, MD may achieve a much better effective coverage, especially when the sensor density is high. This may lead to improved information capture

globally, although higher-threat PoIs may no longer receive proportionally higher allocations of the constrained sensing resources. MDO gives the best balance between matching, fairness, and effective coverage among all the coordination approaches we studied. Its only drawback, which also applies to MD, is the requirement of redistributing the responsibility among all of the sensors when new sensors are added or old sensors are no longer functioning, and the ability to distribute such information to every participating sensor.

2.4.3 Performance of the Stateless Mobility Algorithm

In this section, we first analyse the performance of the Steepest Descent Algorithm to locate the optimal set of parameters of the transition matrix to achieve the best performance according to the defined cost function. We then evaluate the performance of the Markov chain-driven stateless mobility algorithm when the computed optimal transition matrix is used, and contrast the value with the one computed from the Steepest Descent Algorithm.

Using Steepest Descent Algorithm to Determine the Optimal Transition Matrix

In this section we study the performance of the steepest descent algorithm. For simplicity of exposition, we consider the case that the α_i 's and β_i 's in Eq. (2.14) are all equal, i.e., $\alpha_1 = \alpha_2 = \dots = \alpha_M = \alpha$ and $\beta_1 = \beta_2 = \dots = \beta_M = \beta$. We further define the coverage-time deviation ΔC as

$$\Delta C = \sum_{i \in S} \sum_{j,k} \pi_j p_{jk} (T'_{jk,i} - \Phi_i T'_{jk})^2, \quad (2.17)$$

and the average exposure time \bar{E} as

$$\bar{E} = \sqrt{\sum_{i \in S} \frac{\sum_{j=i} p_{ij} R_{ji}}{1 - p_{ii}}}. \quad (2.18)$$

Hence, the cost function in Eq. (2.9) can be rewritten as

$$U = \frac{1}{2}\alpha\Delta C + \frac{1}{2}\beta\bar{E}^2. \quad (2.19)$$

Note that (i) we use the ΔC defined above instead of $\sum_i(\bar{C}_i - \Phi_i)^2$ (see Eq. (2.7)) as it has an easier computational expression. Qualitatively, both of them measure the discrepancy between the actual and desired coverage profiles; (ii) the quantity \bar{E} is related to the long time average exposure times as $\bar{E} = \sqrt{\sum_i \overline{E_i^2}}$ (see Eq. (2.8)).

We will study the following performance aspects of the steepest descent algorithm.

Trade-off between coverage-time and exposure-time metrics. We examine the characteristics of the steady state distribution of the sensor's locations as we vary the weight of the coverage-time deviation ΔC (Eq. (2.17)) and that of the average exposure time \bar{E} (Eq. (2.18)), i.e., α and β respectively in Eq. (2.19).

Stability and adaptivity of the algorithm. We evaluate the cost U (Eq. (2.19)) of the transition matrix in terms of the *iteration number* of the steepest descent algorithm. (Recall that the algorithm works by iteratively updating the transition matrix.) We also compare the U value achieved by the basic algorithm and its variants given in Section 2.3.2, and test if the algorithm gets stuck at a local minimum.

Performance of actual simulation of the coverage schedules. We verify the performance of steepest descent by applying the computed transition matrices to control the movement of the mobile sensor in the Markov chain simulations. The sensor in a simulation picks the destination of its next transition according to the transition matrix, and pauses at the destination for a fixed time interval before its next transition. We measure ΔC in Eq. (2.17) and \bar{E} in Eq. (2.18).

In our experiments, we use the four topologies shown in Fig. 2.12. Each PoI i in the topologies is the center of the cell labeled with i , and its targeted share of coverage time Φ_i is in parentheses. We assume that in traveling from i to j , the sensor uses the straight-line path between i and j .

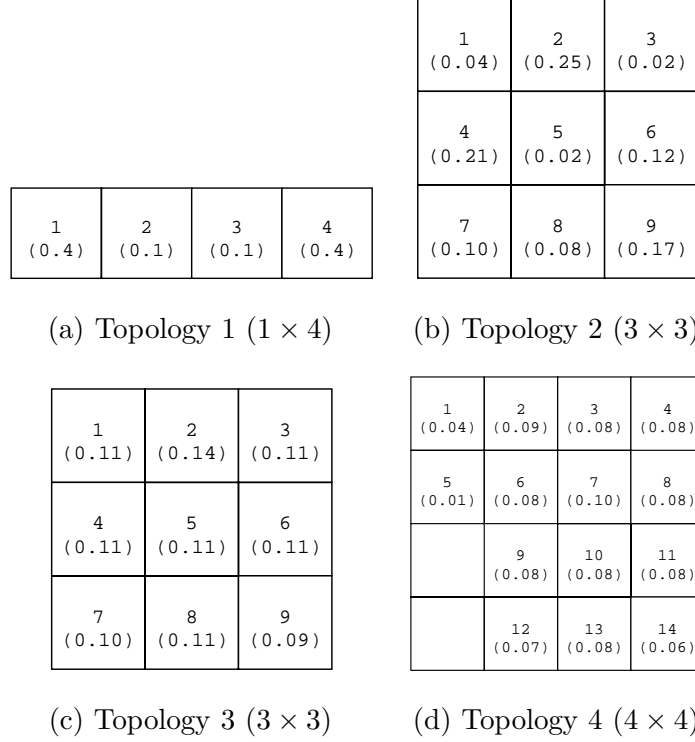


Figure 2.12. Target per-PoI shares of coverage times Φ_i in four simulation topologies.

Existence of Numerous Local Optima

We characterize the search space by comparing the adaptive algorithm (i.e., the basic algorithm modified by variants V2 and V3 in Section 2.3.2) and the perturbed algorithm (i.e., the basic algorithm modified by variants V2–V4 in Section 2.3.2) under Topology 3 in Fig. 2.12(c). In computing the cost function, we illustrate the cases of considering (i) \bar{E} only ($\alpha = 0, \beta = 1$) and (ii) both ΔC and \bar{E} ($\alpha = 1, \beta = 1$), while having $\epsilon = 0.0001$ and $k = 10000$. The CDFs of the achieved costs U_ϵ by the adaptive and perturbed algorithms are shown in Figs. 2.13(a) and 2.13(b) for the cases (i) and (ii), respectively.

From the figures, observe that the adaptive algorithm hits a large number of local optima in its search, from which it is not able to discover a better cost U in the computed descent direction. The presence of numerous local optima makes it essential

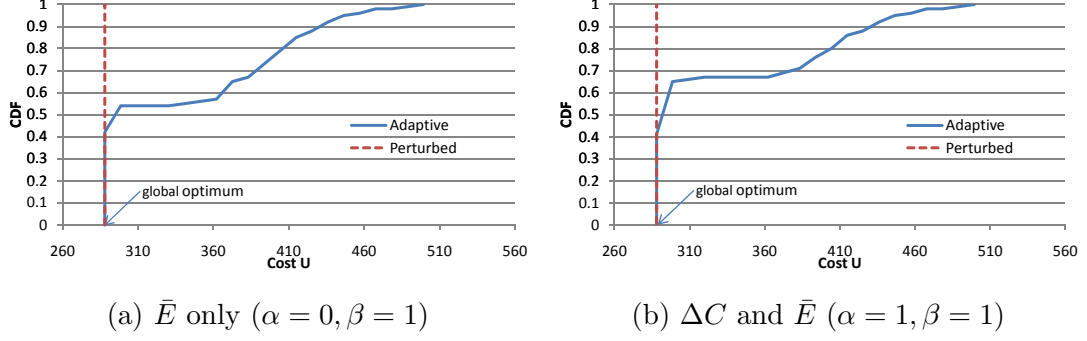


Figure 2.13. CDFs of achieved cost U_ϵ in a large number of runs by the adaptive algorithm and the perturbed algorithm (**Topology 3**).

to adopt noise in the search process, as detailed by the design of the perturbed algorithm in Section 2.3.2. From Fig. 2.13, notice that the perturbed algorithm achieves much better performance than the adaptive algorithm, as evidenced by the extremely sharp rise of the CDF at the global optimum solution. *Indeed, the perturbed algorithm computes a solution extremely close to, if not exactly the same as, the global optimum in all the runs of the experiment, irrespective of the initial search parameters.*

Trade-off between ΔC and \bar{E} Metrics

We study the characteristics of the transition matrix $\{p_{ij}\}$ and the steady-state distribution π_i of the Markov chain, when we vary α and β in Eq. (2.19), using Topology 1 in Fig. 2.12(a). We use $\epsilon = 0.0001$ and $\Delta t = 0.000001$, and vary β from 1 to 0.0000001 while keeping $\alpha = 1$. We also study two extreme cases: (i) $\alpha = 1$ and $\beta = 0$ (we are concerned with the ΔC metric only); and (ii) $\alpha = 0$ and $\beta = 1$ (we are concerned with the \bar{E} metric only). The results for the \bar{C}_i and \bar{E}_i (defined in Eq. (2.7) and (2.8)) are shown in Tables 2.3 and 2.4, respectively.

Observations. From the simulation results, we deduce that when we reduce β , \bar{C}_i (the fraction of time PoI i is actually covered) will more closely approximate the target share of coverage time, while \bar{E}_i (the average exposure time of i) grows, as shown in Tables 2.3 and 2.4. Hence, ΔC decreases while \bar{E} increases. It is because

Table 2.3
 \bar{C}_i for $\epsilon = 0.0001$, $\Delta t = 0.000001$, **Topology 1**.

$\alpha : \beta$	\bar{C}_i			
	1	2	3	4
0:1	0.214	0.286	0.286	0.214
1:1	0.250	0.250	0.250	0.250
1:0.1	0.250	0.250	0.250	0.250
1:0.01	0.254	0.246	0.246	0.254
1:0.001	0.316	0.183	0.183	0.317
1:0.0001	0.369	0.130	0.131	0.370
1:0.00001	0.394	0.107	0.106	0.393
1:0.000001	0.399	0.101	0.101	0.399
1:0	0.4	0.1	0.1	0.4

Table 2.4
 \bar{E}_i for $\epsilon = 0.0001$, $\Delta t = 0.000001$, **Topology 1**.

$\alpha : \beta$	\bar{E}_i			
	1	2	3	4
0:1	9.001	9.001	9.001	9.001
1:1	8.992	9.040	9.004	8.960
1:0.1	8.922	9.082	9.082	8.922
1:0.01	8.255	9.836	9.835	8.254
1:0.001	15.811	24.502	24.398	15.703
1:0.0001	29.770	62.513	62.099	29.524
1:0.00001	43.062	104.126	105.010	43.657
1:0.000001	48.264	118.937	118.898	48.290
1:0	6.944	13742.767	13742.768	6.944

upon reducing β , we are less concerned about the exposure time of the PoI, and are more focused on the target coverage time. We can also observe that \bar{C}_i and \bar{E}_i are

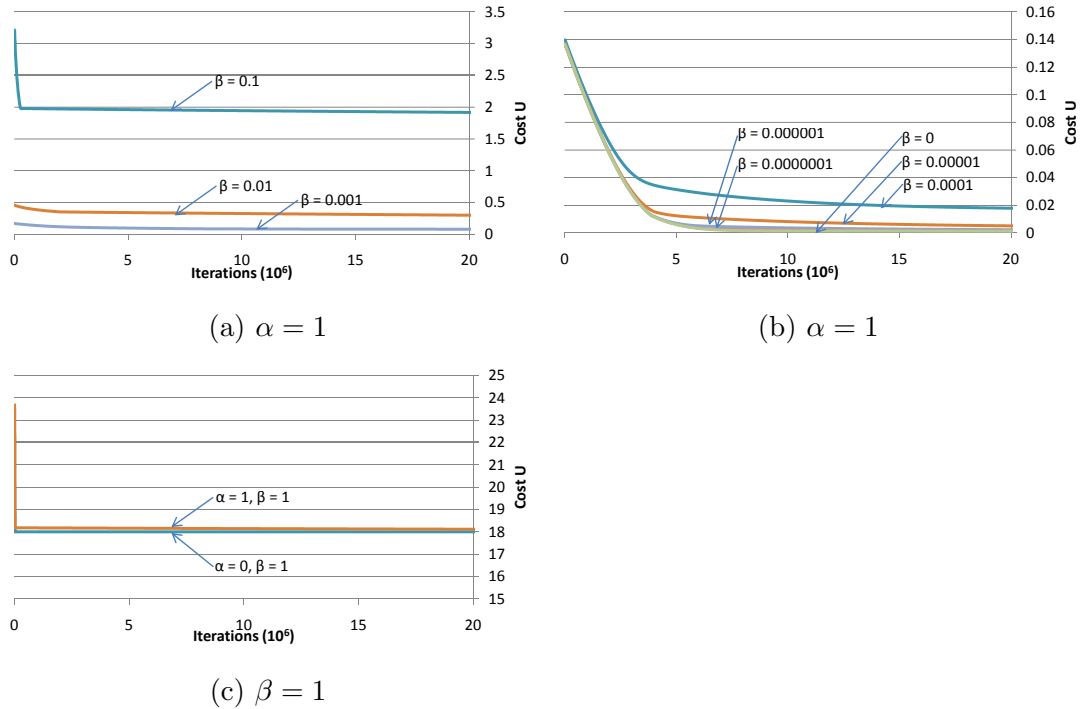


Figure 2.14. Cost function value U in terms of the iteration number.
 $\epsilon = 0.0001$, $\Delta t = 0.000001$, **Topology 1**.

not significantly changed when β is large. It is because for Topology 3, the magnitude of \bar{E} is significantly larger than ΔC . Hence, even if β (weight of \bar{E}) is reduced from 1 to 0.01, the exposure-time component still dominates the cost function, and the resulting p_{ij} and π_i do not change significantly. We notice that as the size of the topology grows, \bar{E} grows, and we will need to further reduce β before the coverage-time component in U will have an observable effect on the p_{ij} and π_i .

Adaptivity and Stability of the Algorithm.

In this set of experiments, we trace the evolution of the cost function U as the transition matrix changes in each iteration of the steepest descent algorithm. We verify that steepest descent can progressively improve the transition matrix by reducing U over the iterations. To demonstrate that the algorithm does not get stuck at a

Table 2.5

The minimum, maximum, and average optimal cost determined by the adaptive algorithm and the perturbed algorithm ($\alpha = 0$, $\beta = 1$, **Topology 3**).

Algorithm	min U	max U	avg U
Adaptive	288.0125	499.3528	332.3202
Perturbed	288.0125	288.0145	288.0127

local minimum, we show convergence to the same transition matrix and U value from different initial p_{ij} 's using the perturbed algorithm in Section 2.3.2.

We study the stability and adaptivity of the algorithm under three settings ($\epsilon = 0.0001$, $\Delta t = 0.000001$):

(1) ΔC only ($\alpha = 1$, $\beta = 0$). We evaluate the case when the cost function considers the coverage-time deviation metric only. The results of the basic algorithm using Topology 2 are depicted in Fig. 2.16(a). In addition, we evaluate the perturbed algorithm with different initial values of p_{ij} 's (i.e., the initial p_{ij} 's are generated using different random seeds), and the results are shown in Fig. 2.16(b).

(2) \bar{E} only ($\alpha = 0$, $\beta = 1$). We evaluate the case when the cost function considers the exposure-time metric only. The results of the basic algorithm are depicted in Fig. 2.15. In addition, we evaluate the cases when the adaptive algorithm and the perturbed algorithm (recall the definitions in Section 2.3.2) are used. The minimum, maximum, and average optimal cost determined by the algorithms using Topology 3 of 200 independent runs are summarized in Table 2.5.

(3) Both ΔC and \bar{E} ($\alpha = 1$, $\beta = 1$). We further consider the case the coverage-time deviation metric and the exposure time metric have the same weight (i.e., $\alpha = \beta$). We omit plots of results for the basic algorithm because they are quite similar to the case when we consider \bar{E} only. The results for different α and β values for the basic algorithm using Topology 1 are shown in Fig. 2.14.

Observations. From the simulation results, we deduce that:

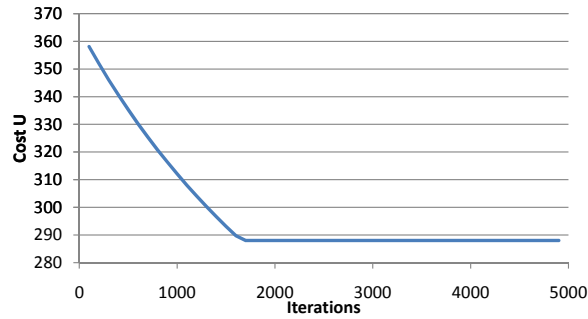


Figure 2.15. Performance of the basic algorithm: cost function value U as a function of iteration number ($\alpha = 0$, $\beta = 1$, **Topology 3**).

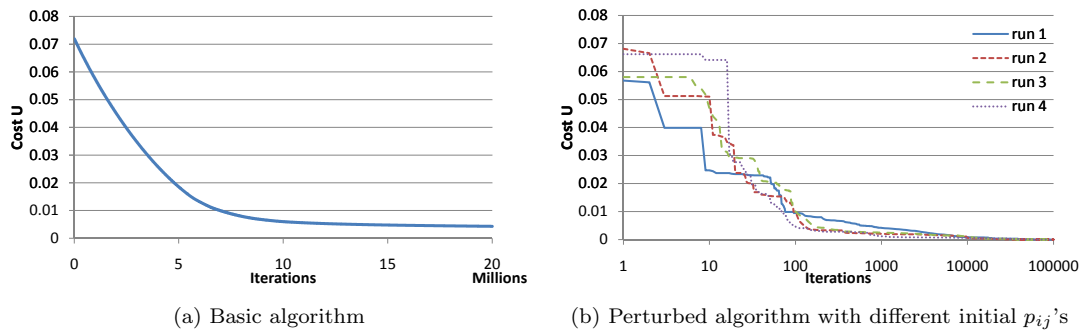


Figure 2.16. Performance of (a) the basic algorithm, and (b) the perturbed algorithm with different initial p_{ij} 's: cost function value U as a function of the iteration number ($\alpha = 1$, $\beta = 0$, **Topology 2**).

1. During the optimization process, the cost function generally decreases until it reaches a stable value as depicted in Figs. 2.14 and 2.16. However, the marginal reduction in the cost becomes smaller as the number of iterations becomes larger.
2. Using different random seeds to generate the initial p_{ij} 's does not affect the performance of the perturbed algorithm (Fig. 2.16(b)) in that it will converge to the same stable values of the cost function. However, the convergence process and its time will be affected. Moreover, the perturbed algorithm can converge extremely close to the same optimal costs in all these situations, i.e., the al-

Table 2.6
Performance for different α/β ratios using Topology 3.

$\alpha : \beta$	ΔC	\bar{E}
0:1	0.0095	288.002
1:1	0.0072	288.010
1:0.0001	0.0052	288.974
1:0	0.0012	1143.45

gorithm is not trapped at a local minimum. This result also holds for more complex topologies.

3. The best solution by the adaptive algorithm (i.e., without noise in the optimization) can have a large range depending on where the search starts. For instance, Table 2.5 shows that the difference between the minimum and maximum returned best costs by the adaptive algorithm is much larger than that by the perturbed algorithm. The average performance of the perturbed algorithm is also much better than that of the adaptive algorithm. The difference in performance is due to the adaptive algorithm being trapped at one of the numerous local optima in the solution space, and the perturbed algorithm's ability to jump out of the local optima.

Performance of actual Markov chain simulations

We evaluate the performance of actual sensor schedules as they are controlled by the Markov chain with the transition matrices found by the steepest descent algorithm. We verify that the cost function computed by steepest descent indeed reflects the realized ΔC and \bar{E} metrics when the computed transition probabilities are applied in practice.

We vary the weights α and β in Eq. (2.14) as before, and use steepest descent to compute the optimal transition matrix. A matrix generated by each iteration

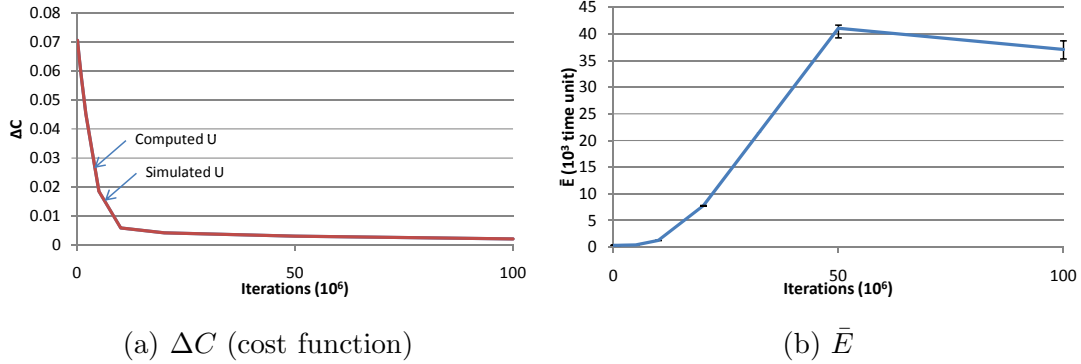


Figure 2.17. Performance of coverage algorithm: (a) ΔC and (b) \bar{E} as a function of the iteration number ($\alpha = 1$, $\beta = 0$, **Topology 2**).

of the steepest descent algorithm is used to drive a corresponding Markov chain simulation of the mobile sensor schedule, and the ΔC and \bar{E} values are measured. Each simulation is repeated ten times to obtain the reported average. 25-th and 75-th percentiles of the measured values are reported as error bars where they are significant. Combinations of the α and β used in different cases are as follows (again $\epsilon = 0.0001$, $\Delta t = 0.000001$):

(1) ΔC only ($\alpha = 1$, $\beta = 0$). The cost function considers only the coverage-time deviation measure. The results are shown in Figs. 2.17 and 2.18 for Topology 2 and Topology 4, respectively.

(2) \bar{E} only ($\alpha = 0$, $\beta = 1$). The cost function considers the exposure-time metric only. As the steepest descent algorithm stabilizes very quickly, we only evaluate the stabilized transition matrix in the sensor simulations, and the results are shown in Table 2.6.

(3) Both ΔC and \bar{E} ($\alpha = 1$, $\beta = 0.0001$). We study the case in which we use a small β . The results are shown in Fig. 2.19 for Topology 3.

Observations. From the simulation results, we deduce that

1. When $\beta = 0$, the measured U in the simulations gives a perfect match with the U value computed by steepest descent (Figs. 2.17 and 2.18). If exposure time is also considered ($\beta > 0$), the measured U in the simulations gives a very close

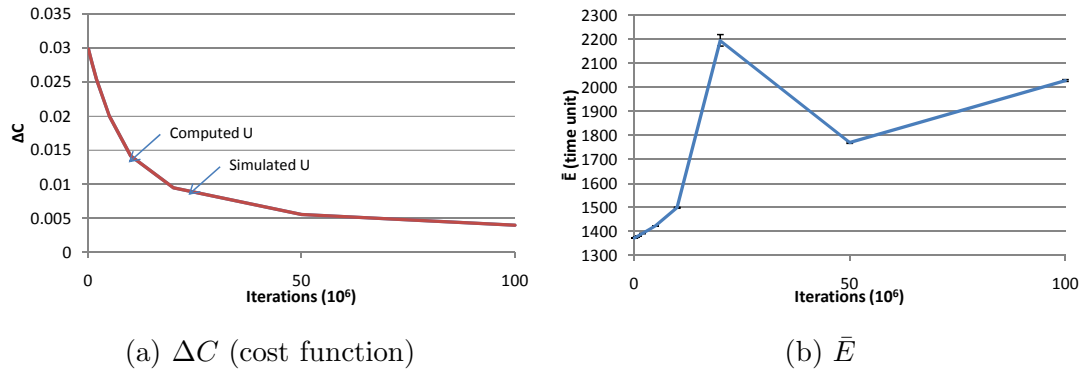


Figure 2.18. Performance of algorithm: (a) ΔC and (b) \bar{E} as a function of the iteration number ($\alpha = 1$, $\beta = 0$, **Topology 4**).

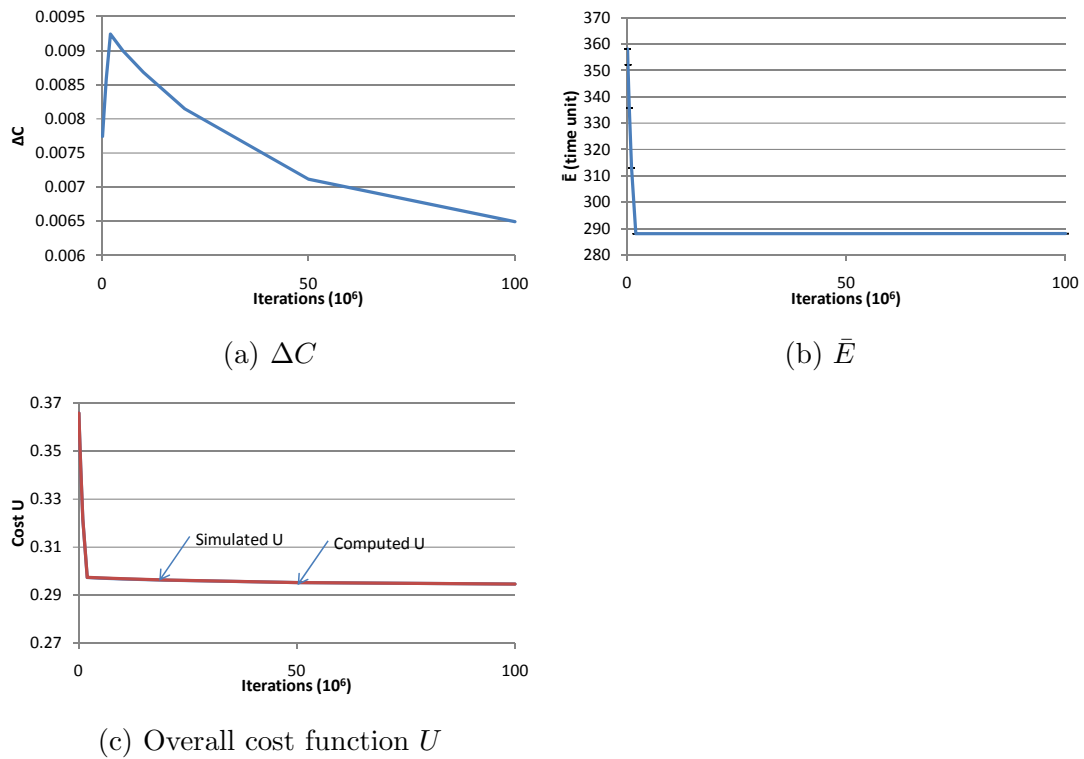


Figure 2.19. Performance of algorithm: (a) ΔC , (b) \bar{E} , and (c) overall cost function U as a function of iteration number ($\alpha = 1$, $\beta = 0.0001$, **Topology 3**).

match with the computed U by the steepest descent (Fig. 2.19). However, the match is not exact because in computing \bar{E} , the analytical formula makes the

simplifying assumption that the time duration of each transition is the same, which does not hold in practice. However, observe from Fig. 2.19 that the difference between the simulated and computed values is very small.

2. If $\beta = 0$, then \bar{E} becomes big, but it does not necessarily worsen as the ΔC metric improves (Fig. 2.17). Moreover, the extent to which \bar{E} worsens is not determined by the size of the map, but the target allocation of per-PoI coverage times (compare Figs. 2.17 and 2.18).

2.5 Summary

In this chapter we have presented two stochastic mobility algorithms which give good performance in terms of network coverage properties. In the stateful mobility algorithm, WRW-aLP, we showed analytically and experimentally the use of the pause time parameter, P , to achieve a balance between matching the expected proportional share of resource and fairness, which are antagonistic in nature. We also studied different coordination approaches when multiple sensors are deployed, and suggested the use of a static division of responsibility among the sensors to achieve a good balance between matching, fairness, and coverage overhead. In the stateless mobility algorithm, we presented (1) the use of Markov chain to model the goals of the sensor network, namely matching and fairness, into a cost function, and (2) the use of steepest descent algorithm to determine the optimal parameters for the transition matrix to achieve the best cost. From the experiments we discovered the possibility of numerous local optimal in the search space, and we suggested the use of stochastic perturbation to help us jump out from the local optimal.

3 QUALITY OF EVENT MONITORING BY MOBILITY ALGORITHMS

As suggested in our motivation in Chapter 1, it is not enough to focus on the coverage properties of mobile sensor networks only because they are used to monitor events, which are mostly uncertain in nature. In this chapter, we begin by giving a comprehensive analytical and experimental study on the quality of monitoring by mobile sensors following a deterministic periodic mobility schedule with movements of different granularity and proportional share at a cell [31]. We then present the event monitoring performance of the mobility algorithm, WRW-aLP, we proposed in Chapter 2 [32].

3.1 Problem Formulation

We assume that events appear and disappear at given points of interest (PoIs) and are to be monitored by sensors whose sensing range is R and whose sensing region is a circle of radius R . This widely applied “perfect disk” sensing model is a simplifying assumption as the coverage regions of real-world signals have been found to be non-isotropic [33]. While it is possible to obtain more accurate numerical solutions by considering more elaborate sensing models (e.g., accounting for the exact geometry of the sensing region or specifying the sensing range as a random variable of some probability distribution), the simple model allows us to obtain essential results about how event monitoring is impacted by the event types and dynamics, without being detracted by the more involved mathematics. We assume the PoIs are located on a 2D plane. A pair of PoIs, say i and j , are connected by a road, given by E_{ij} , of distance d_{ij} . If there is no road that directly connects i and j , $d_{ij} = \infty$. Otherwise, the sensor traveling at speed v from i to j takes time d_{ij}/v to complete the trip.

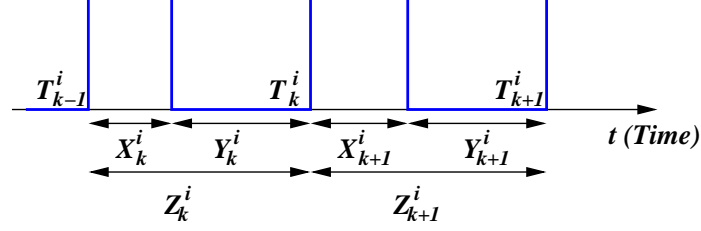


Figure 3.1. Event dynamics at PoI i , with the event staying and absent times X_k^i and Y_k^i for $k = 1, 2, \dots$

The next set of assumptions concerns the event dynamics. The events appear at PoI i one after another. After appearing, each event stays for a duration of time, which we call the *event staying time*, and then disappears. The next event appears after another duration of time, which we call the *event absent time*. Here and in the following, we will use superscript and subscript to denote the PoI and event indices. We denote the sequential staying and absent times by $\{X_k^i\}_{k \geq 1}$ and $\{Y_k^i\}_{k \geq 1}$. The *event inter-arrival time* is then denoted by $Z_k^i = X_k^i + Y_k^i$. We assume that (for each i) the vectors $\{(X_k^i, Y_k^i)\}_{k \geq 1}$ are i.i.d. random variables drawn from a common distribution (X^i, Y^i) , even though for each k , the X_k^i and Y_k^i may be dependent. The commonly known *event arrival times* can be recovered by the formula: $T_0^i = 0$, $T_k^i = T_{k-1}^i + Z_k^i$ for $k, i \geq 1$. These variables are illustrated in Fig. 3.1.

An important assumption behind the analysis of the current formulation is that the event dynamics at different PoIs are independent. This is justified in two situations: (i) there are indeed no correlations between the PoIs because they are physically isolated or are far apart relative to the spatial extents of the events; and (ii) regardless of the presence or absence of correlations between the PoIs, the information acquired at different PoIs is not aggregated but is accounted for on a *per-PoI basis*. Clearly, if information can be aggregated across PoIs to gain further global information about a target (e.g., movement of the target in a global surveillance area), the above assumption will have to be relaxed, but such aggregation is beyond the scope of the current formulation.

We further classify the events as follows. When the staying time draws from X^i is an infinitesimally small amount of time ϵ , the corresponding events are like “blips,” i.e., they do not stay but disappear instantaneously after arrival. Another type of events is that which stays, i.e., there is an $0 < \epsilon \ll 1$ such that $P(X \geq \epsilon) = 1$.

An event at a PoI is captured by the sensor provided that the PoI is within range of the sensor during the event’s lifetime. We assume that events are *identifiable*, i.e., the sensor knows if it is observing the same event or not at different visits to a cell. It is important to note that many investigative actions in the real-world assume the identifiability of events naturally, either by reasonable domain knowledge or because the target possesses easily identifiable attributes. On the other hand, if events are not identifiable, then it is not possible to accumulate knowledge across non-contiguous measurements and the information learned for an event will be in principle the *maximum* obtained for that event among all the individual measurements. However, in practice, the set of measurements that should be used in taking the maximum is not clear without event identities. The monitoring of such non-identifiable events is beyond the scope of the current formulation. We assume that when the sensor observes an event, the information it accumulates about the event is non-decreasing as the observation time increases.

3.1.1 Utility Functions

To quantify the quality of monitoring by the mobile sensors, we propose the use of five different utility functions as depicted in Fig. 3.2. We believe these five utility functions represent how confidence about a captured event grows as a function of time in most, if not all, of the scenarios where mobile sensors are deployed. The significance of these utility functions are discussed as follows,

- *Step utility function*, $U_I(x) = 1$ for $x > 0$. Full information of an event is captured upon contact. This utility function models the capturing of simple events which can be identified immediately upon entering the sensing range of

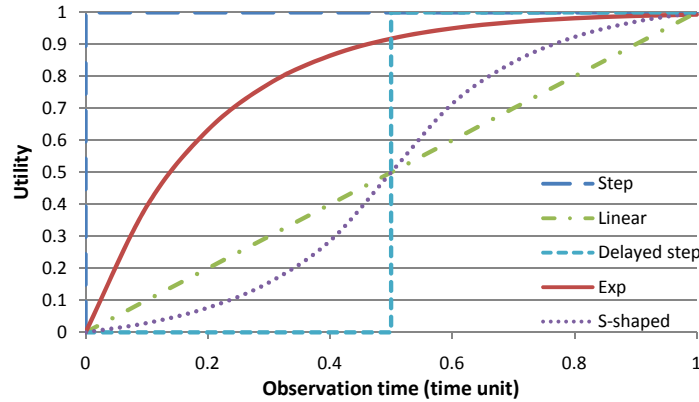


Figure 3.2. The utility functions.

the sensor. This utility function is also widely used in the research community when event capturing ability of a mobility algorithm is to be measured.

- *Exponential utility function*, $U_E(x) = 1 - e^{-Ax}$. Information gain of a captured event is large initially, but the marginal gain diminishes as capture time increases. This utility function models events with confidence following the law of diminishing marginal returns, such as the detection of radioactive sources as depicted in Fig. 3.3, which shows the fraction of averaging values that falls within a range of the actual radiation strength using different window size, w , reported by a radiation sensor to detect a low-strength radiation source, and the approximation using the exponential utility function.
- *Linear utility function*, $U_L = Mx$ for $0 \leq x \leq \frac{1}{M}$, and $U_L = 1$ for $x > \frac{1}{M}$. Information gain of a captured event grows linear with the capturing time. This utility function models events whose confidence grows linearly with its capturing time, and each measured value is equally important towards the overall knowledge of the captured event.
- *S-shaped utility function*, U_S . Information gain increases exponentially initially until a critical observation time is reached, then the marginal gain diminishes

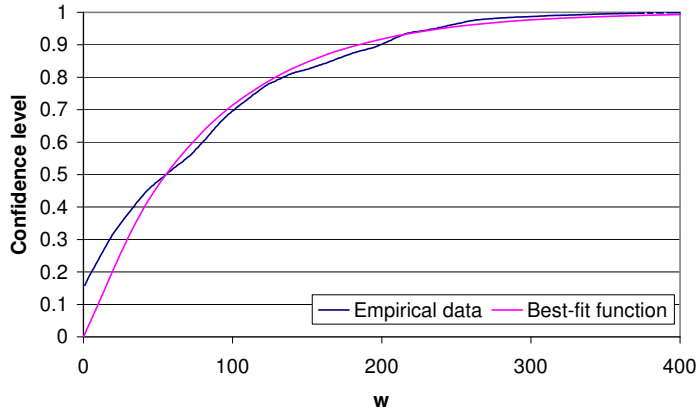


Figure 3.3. Utility of measurement as a function of w : Empirical characterization and least-square fit function.

as capture time further increases. This utility function models general learning trends which are combinations of positive and negative learning curves [34].

- *Delayed-step utility function*, $U_D(x) = U_I(x - D)$. No information is gained by capturing the event until a critical observation time is reached, then all information is obtained. This utility function also serves as an approximation of the mathematically less tractable S-shaped utility function for analysis.

When PoI i falls within the range of the sensor, we say that the sensor is *present* at i . Otherwise, the sensor is *absent* from i . Since we are interested in the resource competition between different PoIs in the analysis of the deterministic periodic schedule, we make the following assumption.

Assumption 1 *The PoIs and the roads between them are separated such that (1) no two PoIs fall within the range of the sensor at the same time; (2) for the sensor traveling from PoI i to PoI j on E_{ij} at speed v , i will be within range of the sensor for R/v time before the sensor leaves i , and j will be within range of the sensor for R/v time until the sensor reaches j , and (3) no PoI other than i and j falls within the range of the sensor during the trip on E_{ij} . In general, however, the sensor can*

vary its speed while traveling on a road, and the sensor may cover other PoIs as the sensor travels from the source PoI to the destination PoI.

Notice that Assumptions (1) and (2) also hold in the event monitoring analysis of WRW-aLP, but we do not make Assumption (3) in that case.

3.1.2 Definition of QoM

We now define the quantitative measurement of the QoM at a PoI and for the whole protected area. In the course of a deployment, $e_1^i, \dots, e_{l_i}^i$ denote the sequence of events appearing at PoI i over the duration $[0, T]$ of the deployment. For the event e_j^i , assume that it is within range of the sensor for a total (but not necessarily contiguous) amount of time t_j^i , where $t_j^i \geq 0$. The sensor will then gain a certain amount of information, $U_j^i(t_j^i)$, about e_j^i , where $U_j^i(\cdot)$ is the utility function of e_j^i . The total information gained by the sensor at i is defined by $E_i(T) = \sum_{1 \leq j \leq l_i} U_j^i(t_j^i)$, and the average information gained per event at i during the whole deployment period is then $\bar{E}_i(T) = E_i(T)/l_i$. Similarly, the total information gained by the sensor in the whole deployment is $E_*(T) = \sum_{1 \leq i \leq n} E_i(T)$, where n is the number of PoIs in the protected area. The average information gained per event in the whole deployment is then

$$\bar{E}_*(T) = \frac{1}{\sum_{1 \leq i \leq n} l_i} \left(\sum_{1 \leq i \leq n} l_i \bar{E}_i(T) \right).$$

By means of the strong law of large numbers and renewal theory, $\bar{E}_i(T)$ and $\bar{E}_*(T)$ can be shown to converge to a deterministic number as $T \rightarrow \infty$. Hence, we define the QoM of PoI i and the whole covered area as:

$$Q_i = \lim_{T \rightarrow \infty} \bar{E}_i(T), \quad \text{and} \quad Q_* = \lim_{T \rightarrow \infty} \bar{E}_*(T). \quad (3.1)$$

Furthermore, they are related by:

$$Q_* = \frac{1}{\mu_*} \sum_{1 \leq i \leq n} \mu_i Q_i, \quad (3.2)$$

where $\mu_i = \frac{1}{E(Z)}$ is the mean event arrival rate at PoI i and $\mu_* = \sum_{1 \leq i \leq n} \mu_i$.

Remark. Note that in defining the QoM of PoI i , we should in principle divide not only by the number of events l_i , but also by the maximum possible utility achievable for an event: $\int_0^\infty U(x)f(x) dx$, where $f(x)$ is the *pdf* of the event staying time distribution. The latter may be less than 1 if the events do not stay infinitely long. However, the difference is only by a proportionality constant, and hence, will not affect our comparison results. Unless otherwise stated, we will further assume that all the events at i have the same utility function, and denote this function by $U^i(\cdot)$.

3.2 Related Work

Quality of monitoring metrics in a sensor network have been proposed, e.g., the rate of false positives in a detection problem [35]. The importance of the sensing time in accurately assessing various physical phenomena has been well documented [36]. The need for non-negligible sensing durations to obtain useful information is due to noise in the measurement process and the probabilistic nature of the phenomena under observation. The impact of the sensing time on the information gained is captured by the event utility functions in our problem statement.

The dynamics of real-world events are frequently modeled as stochastic processes. Poisson arrivals are generally accurate characterizations of a large number of independent event occurrences, whose event inter-arrival times are Exponentially distributed. Real-world network/computing workloads have properties that are found to be long-range dependent and follow the Pareto distribution, e.g., the distribution of traffic in a computer network [37, 38]. In a sensor network, the target events may have similar dynamic behaviors. For example, radioactive particles arriving at a Geiger-Müller counter follow a Poisson process [36]; a chemical leak at a facility may occur with a probability, and the leak may persist for a random duration until the chemical has been dispersed; people may arrive at a location and then leave. Our analysis applies to a wide range of event inter-arrival and staying time distributions.

Event monitoring at PoIs by mobile sensors has been studied in [25, 39]. [39] study the problem of finding the minimum number of sensors to estimate the state of processes present in a network with a bounded estimate error covariance. They assume that the measurements made by the sensors are coupled with zero mean white Gaussian noise. The consideration of explicit stochastic events at PoIs in the mobile sensing has been studied in [25]. Our problem in this chapter is quite different from these earlier papers. First, we consider differential coverage of PoIs by proportional sharing whereas they do not. In particular, we analyze the QoM of periodic sensor schedules, as a function of the proportional share q/p and the period p . Such analysis has applications besides mobile coverage, e.g., energy-efficient sensing by periodically switching off a sensor. Second, we consider sensing tasks with the temporal dimension as defined by the event utility function, whereas they either do not consider the events explicitly or focus only on the number of captured events (where an event is captured whenever it falls within the sensing range of a sensor).

3.3 QoM of Periodic Schedules

To better analyse how the quality of event capturing of mobility algorithms varies with the utility function, we analyse the performance of periodic schedule with different temporal granularities subject to the five utility functions we proposed in Section 3.1.1.

3.3.1 Single-PoI Analysis of QoM

This section forms the basis of the analysis of the impact on the QoM by the coverage schedule of a sensor at a given PoI. The schedule specifies the time intervals over which the sensor is present at or absent from the PoI. A given schedule is achieved by how the sensor moves between the PoIs according to some movement algorithm.

We can already illustrate some interesting QoM properties of proportional-share mobile coverage by considering only *periodic* schedules at individual PoIs. Specifically,

we assume that the sensor is alternately present and absent at a PoI, say i , for q^i and $p^i - q^i$ time units, respectively. For example, let S_1 be the following coverage schedule of i :

$$S_1 = \{PAAAPAAA\dots\},$$

for $q^i = 1$ and $p^i = 4$. In the schedule, P denotes one time unit of the sensor's presence and A denotes one time unit of the sensor's absence. Thus the proportional share equals $q^i/p^i = 25\%$ of the sensor's total coverage time.

Clearly, a given proportional share for i can be achieved in many different ways. For example, $q^i = 2$ and $p^i = 8$ give the following schedule S_2 with the same 25% share for i :

$$S_2 = \{PPAAAAAAPPAAAAAA\dots\}.$$

While S_1 and S_2 are equivalent from the proportional-share point of view, they differ in terms of the time scale over which the proportional share is achieved. Specifically, S_1 achieves the 25% share over a time period of 4 time units, whereas S_2 achieves the same share over a period of 8 time units. We say that S_1 has a finer *fairness granularity* than S_2 , and will use p_i to quantify this fairness granularity. Notice that for a fixed proportional share, a smaller p_i implies a proportionately smaller q_i .

The main purpose of this section is to analyze the dependence of the QoM on the utility function and the fairness granularity. As we will focus on a single PoI, the superscript i will be omitted where there is no confusion.

The problem as formulated in Section 3.1 fits perfectly well in the realm of renewal theory (see [40, Chapter 3]). Recall that T_k refers to the k -th event arrival time. Then the function $N([0, t]) = \sum_{k=1}^{\infty} 1_{[0, t]}(T_k)$ is the total number of arrivals in the interval $[0, t]$. Its expectation $m(t) = E(N([0, t]))$ is called the *renewal function*. Many important quantities about the renewal process $\{T_k\}_{k \geq 1}$ can be expressed in terms of $m(\cdot)$. In the following, we use $\mu = 1/E(Z_k)$ to denote the event arrival rate. The main results from renewal theory we will use are:

1. $\lim_{t \rightarrow \infty} \frac{N([0, t])}{t} = \mu$ a.s.;

2. $\lim_{t \rightarrow \infty} \frac{m(t)}{t} = \mu$;
3. $\lim_{t \rightarrow \infty} m(t+a) - m(t) = \mu a$, for any $a > 0$.

The last statement is true provided that the distribution of Z is not lattice. It shows that regardless of the distribution of Z , in the long run, the “probability” of an event arriving in an interval dt equals μdt .

The following two types of event staying time distribution will be considered in this paper, where $f(x)$ is the pdf of X :

- Exponential Distribution ($\lambda > 0$):

$$f(x) = \lambda e^{-\lambda x}, \quad x > 0, \quad \text{mean} = \frac{1}{\lambda}.$$

- Pareto Distribution ($\alpha, \beta > 0$):

$$f(x) = \frac{\alpha \beta^\alpha}{x^{\alpha+1}}, \quad x > \beta, \quad \text{mean} = \frac{\alpha \beta}{\alpha - 1} \quad (\text{for } \alpha > 1).$$

Furthermore, as a simplification for the simulations, the statistics of the event absent times Y_k^i 's is taken to be the same as that for the event staying times X_k^i 's, even though this is by no means necessary.

We first explain the intuition in analyzing the QoM function. The main step in computing the QoM at a PoI is to consider the *overlapping periods* during which both the event and the sensor are present at the same PoI. A complication is that the sensor can leave and come back multiple times to the same PoI and observe the same event. Hence, the *total* observation period of a single event will in general be a collection of *disjoint* time intervals. See Fig. 3.4 for an example.

For the convenience of the presentation that follows, we denote the proportional share $\frac{q}{p}$ by γ . Furthermore, we use $P_j = [(j-1)p, (j-1)p+q]$ and $A_j = [(j-1)p+q, jp]$ to refer to the j -th sensor present and absent periods, respectively. For many of the proofs, it is sufficient to consider just the case $j = 1$, i.e., $P_1 = [0, q]$ and $A_1 = [q, p]$. This is illustrated in Fig. 3.4.

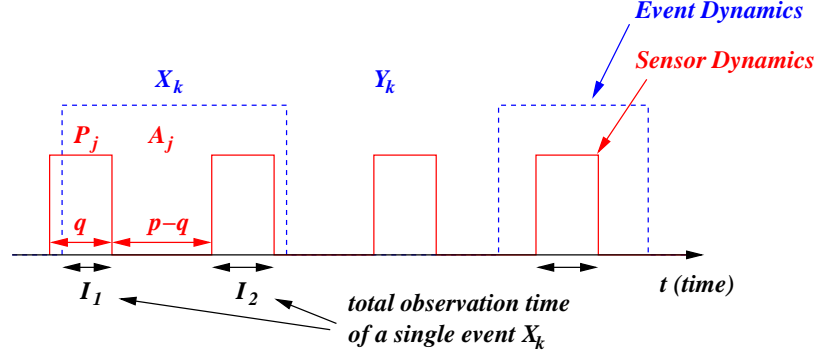


Figure 3.4. Event and sensor dynamics at a PoI. The X_k 's and Y_k 's are the event present and absent periods and the P_j 's and A_j 's are the sensor present and absent periods. The lengths of the P_j 's and A_j 's equal q and $p - q$ respectively. In the above example, the total observation period of the event X_k is the sum of I_1 and I_2 .

As an example, to compute the utility acquired of the event X_k^i in Fig. 3.4, note that the total observation time of the event equals $I_1 + I_2$. Hence, the utility is given by $U(I_1 + I_2)$. This is analytically computable as the statistics of X_k is assumed to be known and the sensor movement is periodic. The machinery of renewal theory is used to handle the statistics of the starting point of the event.

We gradually establish our results and understanding by first considering the step utility function with blip and staying events. Then we write down formulas for general utility functions. Several analytic results are obtained for events with Exponential and Pareto distributions. Now we proceed to present our results.

3.3.2 Step Utility Function

We begin our discussion with events that have the step utility function $U_I(x)$ (see Fig. 3.2). In this case, since the utility reaches one instantaneously, the QoM is equivalent to the fraction of events captured. The next result illustrates the effect on the QoM by a periodic sensor schedule with parameters p and q at a fixed PoI.

Theorem 3.3.1 *For independent arrivals of events that have the step utility function and do not stay, i.e., “blip events,” the QoM at any PoI is directly proportional to its*

share of coverage time q/p . In particular, the achieved QoM does not depend on the fairness granularity.

Proof The statement is a simple consequence of the fact that an event is completely captured if and only if it arrives when the sensor is present. Hence, the QoM is simply the ratio between the expected number (per unit time) of arrivals during the sensor present period q and the total period p , i.e., $\frac{q}{p}$. ■

The above scenario shows that only the proportional sharing information determines the QoM. On the other hand, for events that do stay, the QoM depends on the relationship between the event staying time distribution and the parameters p and q . Specifically, we have the following result.

Theorem 3.3.2 *For independent arrivals of events that stay and have the step utility function, the QoM at a PoI is given by*

$$Q = \frac{q}{p} + \frac{1}{p} \int_0^{p-q} \Pr(X \geq t) dt. \quad (3.3)$$

Proof As the utility function is a step function, the overall utility is given by the total number of events captured when the sensor is present. Note that an event will be captured if (i) it arrives during the sensor present period $[0, q]$; (ii) it arrives during the sensor absent period $[q, p]$, but stays long enough to be captured during the *next* sensor present period $[p, p+q]$. The contribution of (i) to the QoM is given by $\frac{q}{p}$, while that of (ii) is given by $\frac{1}{p} \int_q^p \Pr(X + t \geq p) dt$, which is the second term of Eq. (3.3) after a simple change of variable. ■

Theorem 3.3.2 implies that the sensor that stays at a PoI for $\gamma = q/p$ fraction of the time may be able to capture a significantly *larger* fraction of events than q/p . The following two corollaries give further information about this *extra* fraction of events.

Corollary 3.3.3 *Under the setting of Theorem 3.3.2, with the fairness granularity p kept constant, we have:*

$$\lim_{\gamma \rightarrow 0} Q = \frac{1}{p} \int_0^p \Pr(X \geq t) dt. \quad (3.4)$$

Proof The proof is a direct consequence of Eq. (3.3), upon taking the limit $\gamma \rightarrow 0$. (Note that $q = \gamma p \rightarrow 0$.) ■

This result indicates that no matter how small the proportional share is, there is always some definite, positive gain of information. This is due to the fact that the events stay.

Corollary 3.3.4 *Under the setting of Theorem 3.3.2, the QoM of a given fixed proportional share γ is a monotonically decreasing function of the fairness granularity, i.e., Q decreases as p increases. Furthermore,*

$$\lim_{p \rightarrow 0} Q(p) = 1, \quad \text{and} \quad \lim_{p \rightarrow \infty} Q(p) = \frac{q}{p}. \quad (3.5)$$

Proof The statement again is a simple consequence of Eq. (3.3) which is re-written in the following form:

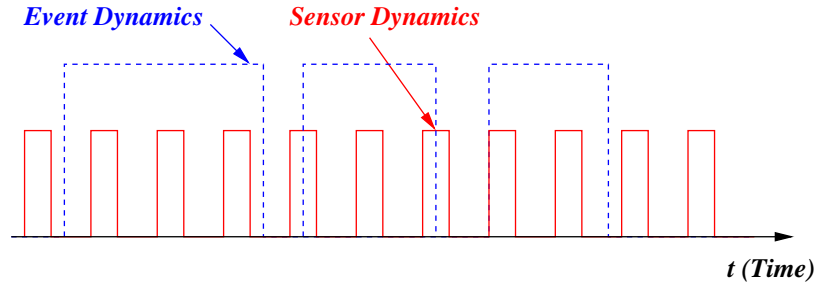
$$Q = \gamma + (1 - \gamma) \frac{1}{(1 - \gamma)p} \int_0^{(1 - \gamma)p} \Pr(X \geq t) dt.$$

Note that the second term in the above is the average over the interval $[0, (1 - \gamma)p]$ of the *monotonically decreasing* function $\Pr(X \geq t)$ of t . Furthermore, $\lim_{t \rightarrow 0} \Pr(X \geq t) = 1$ and $\lim_{t \rightarrow \infty} \Pr(X \geq t) = 0$. Hence,

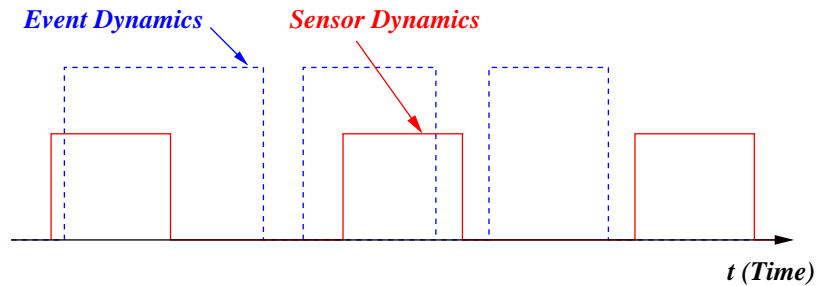
$$\begin{aligned} \lim_{p \rightarrow 0} \frac{1}{(1 - \gamma)p} \int_0^{(1 - \gamma)p} \Pr(X \geq t) dt &= 1 \\ \text{and} \quad \lim_{p \rightarrow \infty} \frac{1}{(1 - \gamma)p} \int_0^{(1 - \gamma)p} \Pr(X \geq t) dt &= 0, \end{aligned}$$

which leads to the stated result. ■

In contrast to Theorem 3.3.1 which applies for blip events, Corollary 3.3.4 implies that finer-grained fairness *does* generally improve the QoM for events which stay and have the Step utility function. In particular, no matter how small the proportional share is, an arbitrarily high QoM can be achieved by an extremely fine fairness granularity. This makes intuitive sense for two reasons. First, by leaving and coming back to a PoI *infinitely often* and *fast*, the sensor can capture any event (which



(a) The advantage of having finer granularity for events that stay.
Every event will be observed with the maximum utility.



(b) The disadvantage of coarser granularity for events that stay.
Some events will be missed while it is wasteful to observe an event for a longer time.

Figure 3.5. Step utility function. Illustration of sensor dynamics versus event dynamics.

stays) while maintaining the desired proportional sharing. Second, if the granularity is coarse, there is a definite amount of time during which the sensor is absent. Thus, some events will be missed while those events which are already observed will not lead to a higher QoM as the *maximum* amount of utility is already achieved by the *first* moment the event is observed. Hence, it is advantageous to leave the PoI and search for *other new* events. The above reasoning is illustrated in Fig. 3.5.

The following are some explicit examples to illustrate Theorem 3.3.2 and Corollary 3.3.4.

1. Exponential Distribution.

$$Q = \gamma + \frac{1}{p} \int_0^{(1-\gamma)p} \int_t^{\infty} \lambda e^{-\lambda x} dx dt = \gamma + \frac{1 - e^{-\lambda(1-\gamma)p}}{\lambda p}, \quad (3.6)$$

which converges to 1 and γ as $p \rightarrow 0$ and ∞ .

2. Pareto Distribution.

When $(1 - \gamma)p \leq \beta$, then $Q = 1$ because any event will always be captured as its duration is at least β time units long. Hence,

$$Q = \gamma + \frac{1}{p} \int_0^{(1-\gamma)p} \Pr(X \geq t) dt = 1. \quad (3.7)$$

When $(1 - \gamma)p > \beta$, then Q is given by

$$Q = \gamma + \frac{1}{p} \int_0^{\beta} \Pr(X \geq t) dt + \int_{\beta}^{\infty} (1 - \gamma)p \Pr(X \geq t) dt ,$$

which equals

$$\gamma + \frac{1}{p} \left[\beta + \frac{\beta^{\alpha}}{(\alpha - 1)} - \frac{1}{\beta^{\alpha-1}} - \frac{1}{((1 - \gamma)p)^{\alpha-1}} \right] . \quad (3.8)$$

The above expression also converges to γ as $p \rightarrow \infty$.

We now consider a scaling result for mobile sensor coverage k out of n PoIs, whose event arrival and departure processes are i.i.d., as k increases. Assume that initially, the sensor performs periodic schedules among k of the n PoIs such that $q^i = \delta$ and $p^i = k\delta$, for $1 \leq i \leq k$, where δ is a unit of time. The following theorem holds.

Theorem 3.3.5 *The expected fraction of events captured is an increasing function of k , the number of PoIs covered.*

Proof According to Eq. (3.2), the overall QoM is given by:

$$\begin{aligned} Q_* &= \frac{1}{n} \sum_{1 \leq j \leq k} Q_j = \frac{1}{n} \sum_{1 \leq j \leq k} \left[\frac{1}{k} + \frac{1}{k\delta} \int_0^{(k-1)\delta} P(X \geq t) dt \right] \\ &= \frac{1}{n} \left[1 + \frac{1}{\delta} \int_0^{(k-1)\delta} P(X \geq t) dt \right] , \end{aligned}$$

which is clearly an increasing function of k . ■

Theorem 3.3.5 provides a formal justification for mobile coverage, namely that the amount of information captured increases as the sensor moves among more PoIs to search for interesting information. This is in addition to the obvious advantage of fairly distributing the sensing resources among the PoIs.

3.3.3 General Utility Function

We now turn our attention to events that have a general utility function $U(\cdot)$. In this case, we have the following QoM result.

Theorem 3.3.6 *For independent arrivals of events at a PoI that have the utility function $U(\cdot)$ and whose event staying time pdf is given by $f(x)$, the achieved QoM equals ($\xi_i = iq - t$, $\eta_i = x + ip - t$):*

$$\int_0^q \int_0^{q-t} U(x)f(x) dx + \sum_{i=1}^{\infty} \int_0^q U(\xi_i + x)f(\eta_i) dx + \sum_{i=1}^{\infty} \int_{-(p-q)}^0 U(\xi_i) f(\eta_i) dx dt \quad (3.9)$$

$$+ \int_q^p \sum_{i=1}^{\infty} \int_0^q U(\xi_i - q)f(\eta_i) dx + \sum_{i=1}^{\infty} \int_q^p U(\xi_i + t) f(\eta_i) dx dt. \quad (3.10)$$

Proof The above formula follows from the fact that the overall utility available for any particular event depends on the *total* length of the intersecting region (which might be discontinuous) during which both the event and sensor are present. The various summands in integral Eqs. (3.9) and (3.10) correspond to the cases that the event arrives when the sensor is present or absent.

If an event arrives at $t \in [0, q]$, i.e., when the sensor is present, then the total utility available from this event is given by ($\xi_i = iq - t$):

$$\int_0^{q-t} U(x)f(x) dx + \sum_{i=1}^{\infty} \int_{x+t=ip}^{x+t=ip+q} U(\xi_i + x + t - ip)f(x) dx + \sum_{i=1}^{\infty} \int_{x+t=ip-(p-q)}^{x+t=ip} U(\xi_i)f(x) dx.$$

In the above, the different integrals correspond to the cases when the event departure time $t + x$ falls in $[t, q]$, $[ip, ip + q]$, and $[ip - (p - q), ip]$ respectively. A change of variable gives Eq. (3.9).

Similarly, if an event arrives at $t \in [q, p]$, i.e., when the sensor is absent, then the total utility available from this event is given by:

$$\begin{aligned} & \sum_{i=1}^{\infty} \int_{x+t=ip}^{x+t=ip+q} U((i-1)q + x + t - ip) f(x) dx \\ & + \sum_{i=1}^{\infty} \int_{x+t=ip-(p-q)}^{x+t=ip} U((i-1)q) f(x) dx. \end{aligned}$$

A change of variable formula then also gives the form of Eq. (3.10). ■

Eqs. (3.9) and (3.10) above can have a complicated analytical form in general, but they are certainly amenable to numerical computation. Nevertheless, we first present two exact analytical results. (Recall $\gamma = \frac{q}{p}$.)

(1) Exponential utility function $U_E(x) = 1 - e^{-Ax}$ and Exponential staying time: $f(x) = \lambda e^{-\lambda x}$.

$$\begin{aligned} Q &= \frac{A\gamma}{A+\lambda} - \frac{1 - e^{-\lambda q}}{\lambda p} + \frac{\lambda(1 - e^{-(A+\lambda)q})}{(A+\lambda)^2 p} \\ &+ \frac{(e^{\lambda q} - 1)^2}{\lambda p e^{\lambda q} (e^{\lambda p} - 1)} - \frac{\lambda(e^{(A+\lambda)q} - 1)^2 e^{-(A+\lambda)q}}{(A+\lambda)^2 p (e^{(Aq+\lambda p)} - 1)} \\ &+ \frac{2(e^{\lambda(p-q)} - 1)}{p} \times \frac{e^{\lambda q} - 1}{\lambda(e^{\lambda p} - 1)} - \frac{e^{(A+\lambda)q} - 1}{(A+\lambda)(e^{(Aq+\lambda p)} - 1)} \\ &+ \frac{(e^{Aq} - 1)e^{\lambda q}(e^{\lambda(p-q)} - 1)^2}{\lambda p (e^{\lambda p} - 1)(e^{(Aq+\lambda p)} - 1)}. \end{aligned} \quad (3.11)$$

Note that the above leads to

$$\lim_{p \rightarrow 0} Q = \frac{A\gamma}{A\gamma + \lambda}, \quad \text{and} \quad \lim_{p \rightarrow \infty} Q = \frac{A\gamma}{A + \lambda}. \quad (3.12)$$

(2) Delayed-step utility function $U_D(x) = U_I(x - D)$ and Exponential staying time: $f(x) = \lambda e^{-\lambda x}$.

When p is very small such that D is an integral multiple of q , i.e., $D = kq$ for $k = 1, 2, \dots$, we have:

$$Q = e^{-\frac{\lambda D}{\gamma}} \gamma + \frac{e^{\lambda(1-\gamma)p} - 1}{\lambda p}. \quad (3.13)$$

On the other hand, when p is very large, specifically, when $q > D$, then

$$Q = e^{-\lambda D} \gamma + \frac{1}{\lambda} - D \frac{1 - e^{-\lambda(p-q)}}{p}. \quad (3.14)$$

It is also interesting to obtain Eq. (3.14) without using Theorem 3.3.1. In order for the sensor to capture an event and gain enough information, the event staying time and the sensor present period should overlap for at least an interval of length D . Based on this, let t be the time that an event occurs and let x be its staying time. Consider the first sensor present and absent periods $[0, q]$ and $[q, p]$. Then the probability of gaining enough information for an event arrived during these two periods is given by

$$\begin{aligned} & \int_0^{q-D} \Pr(X \geq D) dt \\ & + \int_{q-D}^q \Pr(t + x - p + (q - t) \geq D) dt \\ & + \int_q^{q-D+p} \Pr(t + x - p \geq D) dt, \end{aligned}$$

which gives Eq. (3.14) upon dividing by p .

Combining Eqs. (3.13) and (3.14), we have:

$$\lim_{p \rightarrow 0} Q = e^{-\lambda \frac{D}{\gamma}}, \quad \text{and} \quad \lim_{p \rightarrow \infty} Q = \gamma e^{-\lambda D}. \quad (3.15)$$

For the Pareto type events, even though analogous results can be derived, the analytical formula are quite complicated and hard to be put into closed form. But they demonstrate similar qualitative behaviors as seen in the simulation section.

The above analytical results can be intuitively understood in many ways, and are discussed in the following section.

3.3.4 Implications of Theoretical Results

The first three discussion points concern various limiting cases.

(i) Let the fairness granularity p and the proportional share γ be fixed. Then as the event staying time goes to infinity, every event will always be captured and the maximum value 1 for the utility can be achieved. Therefore, the QoM is an increasing function of the mean event staying time. Note that this scenario corresponds to

$\lambda \rightarrow 0$ for the exponential staying time distribution, and $\beta \rightarrow \infty$ for the Pareto distribution.

(ii) In the limit of $p \rightarrow 0$, every event which stays will always be captured. However, the total observation time is only γ fraction of the event's duration. Hence, the average utility achieved is:

$$Q_0 = \int_0^{\infty} U(\gamma x) f(x) dx. \quad (3.16)$$

This result is consistent with the explicit Eqs. (3.12) and (3.15).

The following give an explicit expression of the QoM for the Pareto event staying time distribution.

- With the exponential utility function U_E ,

$$Q_0 = 1 - \alpha \beta^\alpha \int_{\beta}^{\infty} \frac{e^{-A\gamma x}}{x^{\alpha+1}} dx = 1 - \alpha \int_1^{\infty} \frac{e^{-A\gamma\beta x}}{x^{\alpha+1}} dx.$$

- With the Delayed-step utility function U_D ,

$$Q_0 = \begin{cases} 1 & \text{for } D \leq \gamma\beta, \\ \left(\frac{\gamma\beta}{D}\right)^\alpha & \text{for } D > \gamma\beta. \end{cases}$$

(iii) In the limit of $p \rightarrow \infty$, each event, if captured, will essentially be observed for its whole duration. On the other hand, only γ fraction of the events will be captured. Hence, the QoM is given by:

$$Q_\infty = \gamma \int_0^{\infty} U(x) f(x) dx, \quad (3.17)$$

which is also consistent with the explicit Eqs. (3.12) and (3.15).

Again, for Pareto event staying time distribution, we have:

- With the Exponential utility function:

$$Q_\infty = \gamma \left[1 - \alpha \beta^\alpha \int_{\beta}^{\infty} \frac{e^{-Ax}}{x^{\alpha+1}} dx \right] = \gamma \left[1 - \alpha \int_1^{\infty} \frac{e^{-A\beta x}}{x^{\alpha+1}} dx \right].$$

- With the Delayed-step utility function:

$$Q_\infty = \begin{cases} \gamma & \text{for } D \leq \beta, \\ \gamma \left(\frac{\beta}{D}\right)^\alpha & \text{for } D > \beta. \end{cases}$$

The next two discussion points concern the two most important qualitative descriptions of the QoM function.

(iv) For the step and exponential utility functions, the QoMs are *monotonically decreasing* functions of p . This is because both utility functions are *concave* functions of the observation time. Hence, it is advantageous to capture as many *new* events as possible rather than to gain further information for the event which has already been observed. A finer fairness granularity exactly achieves this. More precisely, a very fine granularity can basically capture *every* single event (as the event stays), and each event captured gives the best possible *initial* utility gain *per unit time*. On the other hand, coarser granularity can miss some events while for those events captured, the information captured per unit time is not maximized due to the concavity of the utility function.

The above is certainly consistent with Theorem 3.3.2 for the step utility function which is qualitatively illustrated in Fig. 3.5. Furthermore, This is also explicitly demonstrated by the analytical Eqs. (3.6), (3.7)-(3.8). It is easy to see that their derivatives with respect to p is negative. For Eq. (3.11), this behavior is graphically demonstrated in Fig. 3.6.

(v) However, the key feature is that for certain utility functions, the *maximum* QoM is only achieved at some *intermediate* fairness granularity. We spend a moment to explain this important phenomenon.

The above observation is easiest to explain for the delayed-step utility U_D . In the limit of $p \rightarrow 0$, any event can always be captured. This is essentially the statement of Corollary 3.3.4. However, in order to gain enough information about the event, it is necessary that the event staying time be at least $\frac{D}{\gamma}$ long. This probability is given by $\Pr(X \geq \frac{D}{\gamma})$. When p is positive (no matter how small it is), however, this is not absolutely necessary. In fact, if the event arrives right at the beginning of a sensor

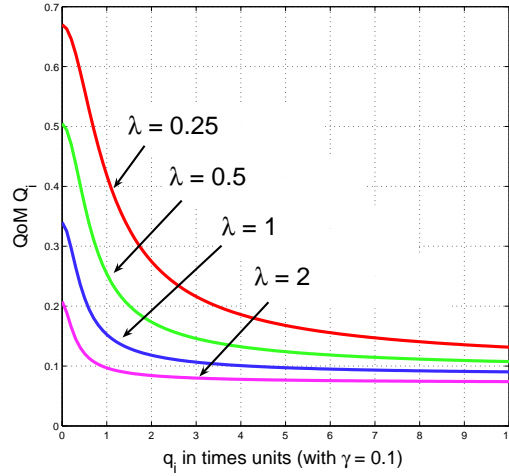


Figure 3.6. The QoM for the exponential utility function and exponential staying time.

present period, then the event staying time just needs to be at least $\frac{D}{\gamma} - (1 - \gamma)p$ long. It is this saving that increases the QoM. Hence, initially the QoM is an *increasing* function of p for *small* p . This can also be seen analytically from Eq. (3.13) by which we have for $0 < p \ll 1$:

$$\text{QoM} \approx e^{-\frac{\lambda D}{\gamma}} \left(1 + \frac{(1 - \gamma)^2 \lambda p}{2} + \dots \right),$$

which is an increasing function of p .

The behavior of QoM when p is *large* is also interesting and in fact quite intricate. From Eq. (3.14), observe that the QoM is a *decreasing*, *constant*, or *increasing* function of p for λ *less than*, *equal to*, or *greater than* $\frac{1}{D}$, respectively. This can be seen that for $q \gg 1$, we have:

$$\frac{d\text{QoM}}{dp} \approx e^{-\lambda D} \left(\gamma - \frac{1}{\lambda} - D \frac{1}{p^2} + \dots \right).$$

The above is due to the competitive effect (for p large) of the *loss* of utility for events arriving near the end of a sensor present period and the *gain* of utility for events arriving before the sensor present period. Hence, for $\lambda < \frac{1}{D}$, the QoM initially increases and then decreases as a function of p , i.e., it is optimal at some *intermediate* p value.

All of the above implications are supported by the simulation results in Section 3.4.

3.3.5 Discussions on Multiple Sensors

In the previous analysis, it is simplest to interpret that the periodic PoI visit schedule is induced by a single periodic sensor. We now discuss an extended scenario in which multiple mobile sensors, each on the same periodic schedule, visit the PoI in sequence. We acknowledge that the general coordination between multiple sensors is a very important problem of practical and research relevance. A full analysis of the scenario deserves another line of work and is out of the scope of the current chapter. On the other hand, we will demonstrate that the present QoM analysis can already lead to some interesting consequences.

We assume that the PoIs $\{L_i\}_{i=1,2,\dots,n}$'s are located along a circular circuit. The mobile sensors $\{P_j\}_{j=1,2,\dots,m}$ move along the circuit in an identical fashion. The geometry of the sensor locations is such that they are clustered together in the following sense (see Fig. 3.7(a)):

$$\begin{aligned} \text{dist}(P_j, P_{j+1}) &= A, \quad \text{for } j = 1, 2, \dots, m-1, \\ \text{dist}(P_m, P_1) &= B. \end{aligned}$$

In the above, the distance is measured in the *clockwise* sense. From each PoI's point of view, the pattern of the coverage time is illustrated in Fig. 3.7(b). As each sensor is associated with a sensing range R , the coverage time is finite, denoted by q . In addition, due to the separations A and B between the sensors, we let the sensor absent times be a and b . We investigate the QoM for each PoI in relation to the following quantities:

1. p : the total period of the sensor movement;
2. m : the number of mobile sensors;
3. $r = \frac{b}{a}$: the "clustering ratio" of the sensors' visit sequence.

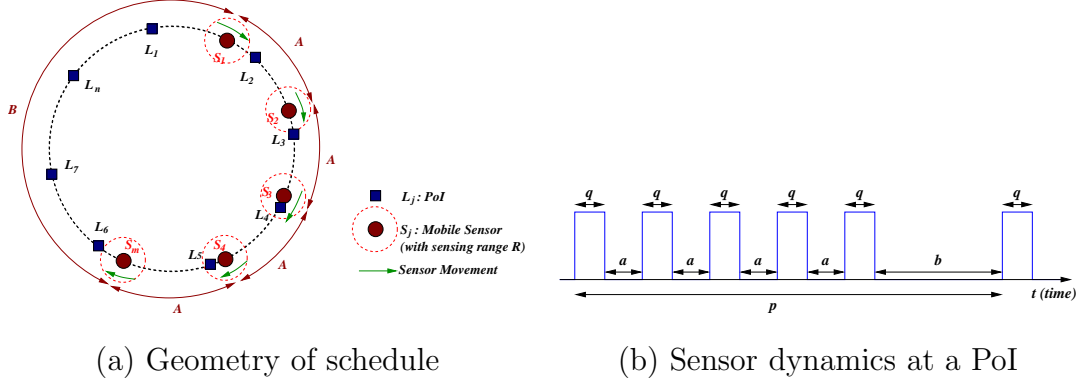


Figure 3.7. Geometry of schedule and sensor dynamics at a PoI using multiple sensors.

Note the relation $p = mq + (m - 1)a + b$. In addition, $r = 1$ implies that the sensors are equidistant from each other while for $r = \infty$, it is equivalent to having a single mobile sensor with coverage time $q' = mq$ for each period p . Furthermore, the information acquired by the sensors is aggregated for possibly higher combined utilities.

First we consider the case of the step utility function. In this case, any event will be captured with the *maximum utility value one* if it occurs when the sensor is present (see Fig. 3.8, event A). On the other hand, if it occurs when the sensor is absent, then the event must stay *at least* till the first moment the sensor comes back in order to be captured (see Fig. 3.8, events B and C). Based on this argument, the QoM function is given by:

$$\text{QoM} = \frac{1}{p} \left[m \int_0^q dt + (m - 1) \int_0^a P(T \geq a - t) dt + \int_0^b P(T \geq b - t) dt \right], \quad (3.18)$$

where T is the event staying time. The pre-factors before the integrals in the above expression come from the fact that there are m identical cases of the event type A , $m - 1$ identical cases of type B , and one case of type C .

To give further explicit analysis, we assume that the event staying time follows the exponential distribution with parameter λ . In this case Eq. (3.18) is reduced to:

$$\text{QoM} = \frac{1}{p} \left[mq + \frac{m - (m - 1)e^{-\lambda a} - e^{-\lambda b}}{\lambda} \right]. \quad (3.19)$$

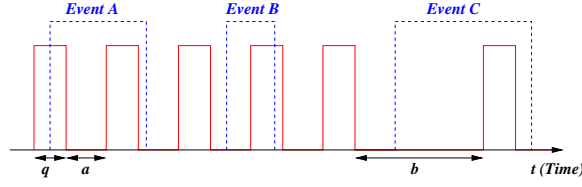


Figure 3.8. The relation between event dynamics and the multisensor coverage pattern. In the above, the number of sensors m equals 5.

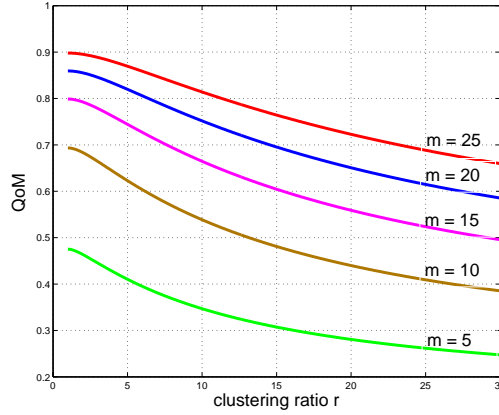


Figure 3.9. Plot of the QoM function (3.20) for the step utility function: $p = 10$, $q = 0.1$, and varying number of mobile sensors m .

In addition, by means of $p = mq + (m - 1)a + b$ and $r = \frac{b}{a}$, we get $a = \frac{p - mq}{m - 1 + r}$. Hence, the above QoM function becomes:

$$\text{QoM} = \frac{1}{p} \left[mq + \frac{m - (m - 1)e^{-\lambda\left(\frac{p - mq}{m - 1 + r}\right)} - e^{-\lambda r\left(\frac{p - mq}{m - 1 + r}\right)}}{\lambda} \right]. \quad (3.20)$$

This QoM function is plotted in Fig. 3.9.

The monotonically decreasing behavior is expected as explained previously. It can also be seen by taking the derivative of the QoM with respect to the clustering ratio r . The result is given by:

$$\frac{d\text{QoM}}{dr} = \frac{(m - 1)(p - mq)}{p(m - 1 + r)^2} \left[-e^{-\lambda\left(\frac{p - mq}{m - 1 + r}\right)} + e^{-\lambda r\left(\frac{p - mq}{m - 1 + r}\right)} \right], \quad (3.21)$$

which is clearly negative for $r \geq 1$.

We now consider the second, more interesting case of the delayed step utility function with delay D . The argument is similar to the previous case, except that in order to gain the maximum utility value one, the event must now stay long enough so that the total observation time is at least D .

To obtain a tractable mathematical formula, we assume that $D = Nq$ for some positive integer N . Then based on the consideration of different event types such as A , B , and C in Fig. 3.8, the QoM function is given by:

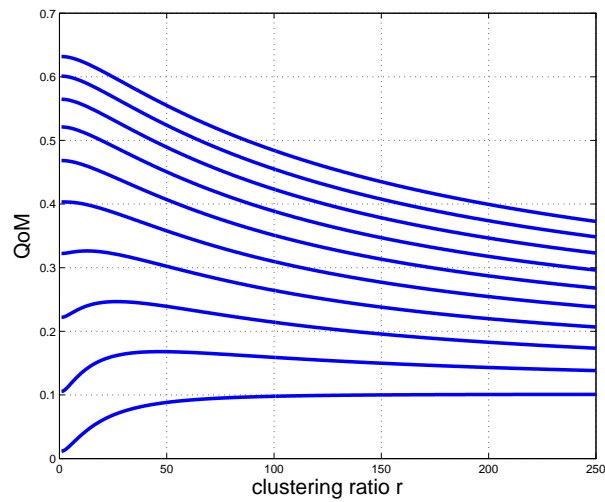
$$\begin{aligned} \text{QoM} = & \frac{1}{p} (m - N) \int_0^q P(T \geq (a + q)N) dt \\ & + (m - N) \int_0^q P(T \geq (a + q)N - t) dt \\ & + N \int_0^q P(T \geq (a + q)N - a + b) dt \\ & + (N - 1) \int_0^a P(T \geq (a + q)N - a + b - t) dt \\ & + \int_0^b P(T \geq (a + q)N - a + b - t) dt . \end{aligned}$$

As before, by assuming the exponential distribution for the event staying time, we have

$$\begin{aligned} \text{QoM} = & \frac{1}{p} \left\{ (m - N)e^{-\lambda(a+q)N} \left[q + \frac{e^{\lambda a} - 1}{\lambda} \right] \right. \\ & \left. + e^{-\lambda(a+q)N+b-a} \left[Nq + (N - 1)\frac{e^{\lambda a} - 1}{\lambda} + \frac{e^{\lambda b} - 1}{\lambda} \right] \right\} . \quad (3.22) \end{aligned}$$

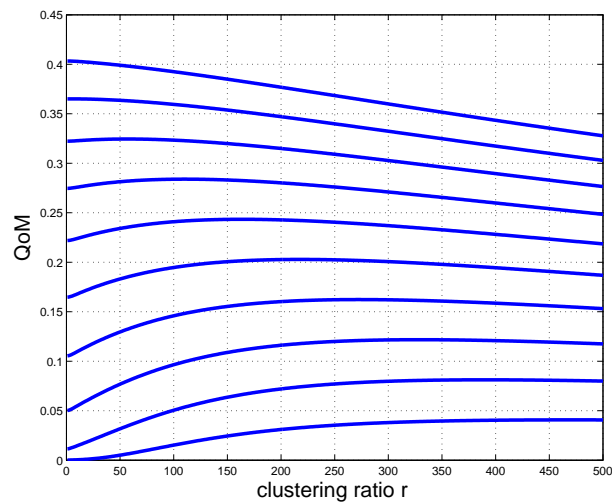
The above function is plotted in Fig. 3.10.

From the above, it appears that the behavior of the QoM is quite elaborate as a function of the clustering ratio r and the number of mobile sensors m . In particular, it can be a *monotonically increasing*, *non-monotonic*, or *monotonically decreasing* function of m , depending on the *competition* between eliminating *redundancies* in the information capture and avoiding the *loss* of events. See also the explicit discussion in Section 3.3.4(v).



(a) $p = 10, q = 0.01, D = 0.05$.

The curves (from below to above) corresponds to $m = 10, 20, 30, \dots, 100$.



(b) $p = 100, q = 0.01, D = 0.05$.

The curves (from below to above) corresponds to $m = 50, 100, 150, \dots, 500$.

Figure 3.10. Plot of the QoM function (3.22).

3.4 Evaluation

We present simulation results to illustrate the analytical results in Section 3.3. Recall the use of X and Y to denote the event staying and absent time variables,

Table 3.1

Maximum available information for capture, averaged over all the simulated events.

Utility function	$X \sim Y \sim \text{Exp}(\lambda)$				$X \sim Y \sim \text{Pareto}(\alpha = 2, \beta)$			
	$\lambda = 0.25$	0.5	1	2	$\beta = 0.25$	0.5	1	2
Step	1	1	1	1	1	1	1	1
Exponential	0.95	0.91	0.83	0.72	0.84	0.97	1	1
Linear	0.89	0.79	0.63	0.43	0.44	0.75	1	1
S-shaped	0.88	0.78	0.62	0.40	0.36	0.82	1	1
Delayed-step	0.88	0.78	0.61	0.37	0.50	1	1	1

respectively. We measure the QoM Q_i achieved over 1,000,000 time units in a simulation run, and report the average Q_i of 10 different runs. The different runs produce results that have extremely small differences. Hence, we omit the error bars in the reported results. Notice that not all the events in a simulation stay long enough to be captured at the full utility. The maximum information available for capture is given by $\int_0^\infty U(x)f(x) dx$ as explained in Section 3.1.2, and the values obtained from the experiments are shown in Table 3.1. Each reported experiment uses the same distribution for both the event staying and absent times, which is either Exponential with varying λ , or Pareto with varying β (and α is kept to be 2).

3.4.1 QoM of Periodic Schedule

In this section we evaluate the QoM performance of the periodic schedule we studied in Section 3.3.

Step utility

We now present results for the step utility function U_I . Figs. 3.11(a) and 3.11(b) show the achieved QoM as a function of the proportional share q/p for Exponential and Pareto event dynamics, respectively. The results agree with Theorem 3.3.2 and

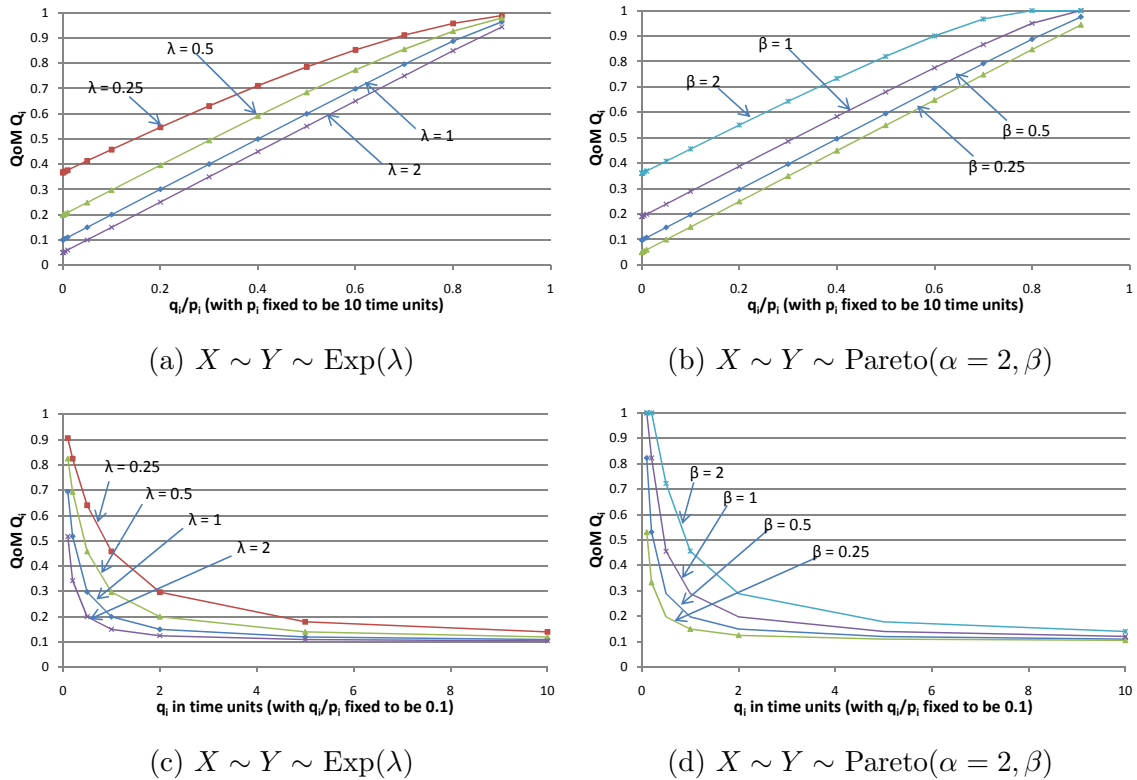


Figure 3.11. Achieved QoM for events that stay and have the step utility function U_I .

its instantiations for the distributions. Note that the fraction of events captured can be significantly higher than the proportional share, e.g., a QoM of close to 0.4 is achieved for $\text{Exp}(\lambda = 0.25)$ and $\text{Pareto}(\alpha = 2, \beta = 2)$ even when the share is only slightly positive (see Corollary 3.3.3). The observation time of the events increases as the events stay longer, and so the QoM is higher when λ is smaller for Exponential event dynamics and β is larger for Pareto event dynamics (see Section 3.3.4(i)). In general, the QoM is not linear in the proportional share.

Figs. 3.11(c) and 3.11(d) show the QoM as a function of the fairness granularity for Exponential and Pareto event dynamics, respectively. As predicted by Corollary 3.3.4, the QoM is a monotonically decreasing function of q (and hence, p , as we have q_i/p_i fixed), meaning that finer-grained fairness will improve performance. As explained before, the QoM increases as λ decreases for the Exponential distribution and as

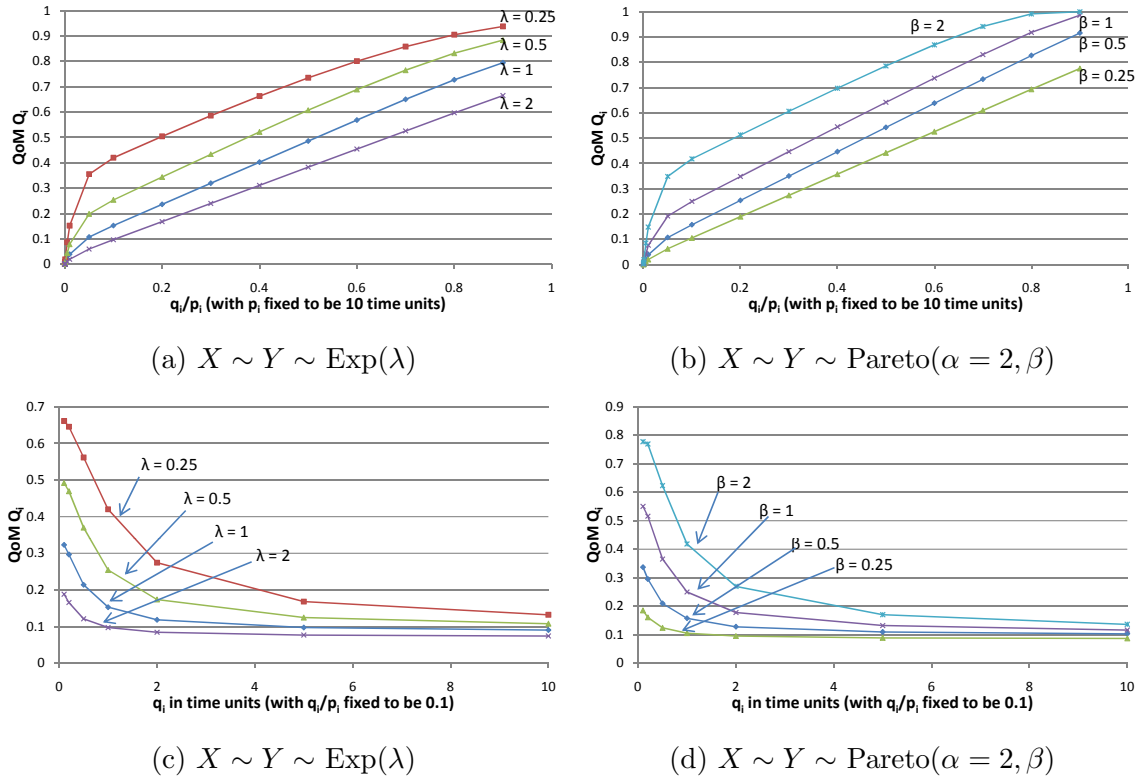


Figure 3.12. Achieved QoM for events that stay and have the exponential utility function U_E , $A = 5$.

β increases for the Pareto distribution. Furthermore, the QoM converges to the maximum value one and the proportional share $\gamma = q/p$ as q (and hence, p) converges to 0 and ∞ , respectively (see Corollary 3.3.4).

Exponential utility

We now present results for the exponential utility function U_E (with $A = 5$). Figs. 3.12(a) and 3.12(b) show the achieved QoM as a function of the proportional share for Exponential and Pareto event dynamics, respectively. Unlike Step utility, the achieved QoM is close to zero when the share is only slightly positive. This is due to the need to accumulate information for Exponential utility. As the share increases initially, however, there is a sharp gain in the QoM. This is because most information is gained during the initial observation of an event for Exponential utility. Moreover,

the initial gain is higher when the events stay longer (i.e., smaller λ or larger β), because longer staying events are more likely to be captured even if they arrive when the sensor is not present. As the share further increases, the marginal gain in the QoM becomes smaller, again mimicking the decreasing marginal gain of information with longer observation time for the type of events. Note that for the larger λ values (e.g., $\lambda = 2$) or smaller β values (e.g., $\beta = 0.25$), the QoM is significantly smaller than one even for a large share. This is in part because at those parameter values, the events do not stay long enough to be captured at their full utility.

Figs. 3.12(c) and 3.12(d) show the achieved QoM as a function of the fairness granularity for Exponential and Pareto event dynamics, respectively. For Exponential utility, the results agree with Eqs. (3.11) and (3.12). (See also Section 3.3.4(iv).) In particular, Fig. 3.12(c) shows that the QoM is monotonically decreasing in q (and hence, p) and gives the correct QoM limits in Eq. (3.12) as $p \rightarrow 0$ and $p \rightarrow \infty$. In addition, the QoM increases when λ decreases. The results in Fig. 3.12(d) show that similar results hold for Pareto event dynamics.

Linear utility

We now present results for the linear utility function U_L (with $M = 1$). Figs. 3.13(a) and 3.13(b) show the achieved QoM as a function of the proportional share for Exponential and Pareto event dynamics, respectively. Figs. 3.13(c) and 3.13(d) show the achieved QoM as a function of the fairness granularity for the two types of event dynamics. The results are similar to Exponential utility. In particular, the QoM is a monotonically decreasing function of p , although it is flat over an initial range of p values, showing that there is no need for the sensor to move faster and achieve a smaller p after some point.

Delayed-step utility

We now present simulation results for the delayed-step utility function U_D ($D = 0.5$ time units). Figs. 3.14(a) and 3.14(b) show the achieved QoM as a function of the proportional share for Exponential and Pareto event dynamics, respectively. They

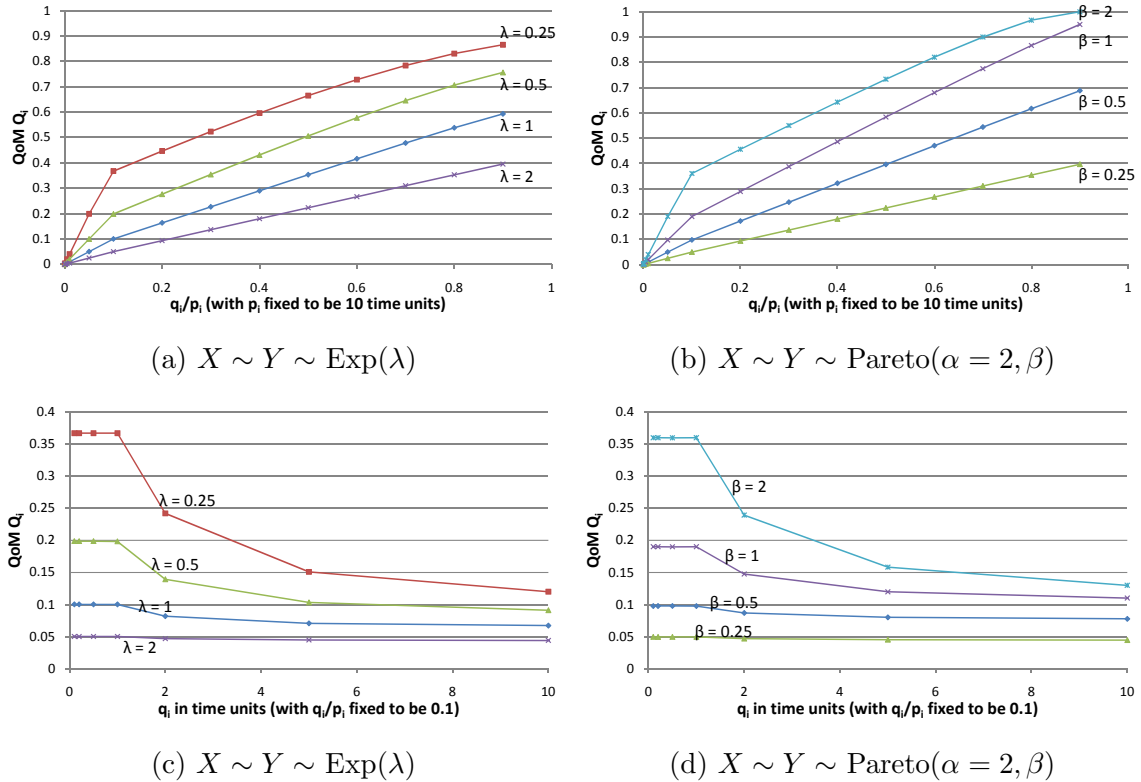


Figure 3.13. Achieved QoM for events that stay and have the linear utility function U_L , $M = 1$.

show that the QoM is monotonically increasing in the proportional share, and the QoM is higher when the events stay longer (i.e., smaller λ or larger β).

Figs. 3.14(c) and 3.14(d) show the achieved QoM as a function of the fairness granularity. Note that in this case, the QoM is no longer monotonically decreasing in p , but the optimal fairness occurs at an intermediate value. Note also that for $\lambda = 2 = \frac{1}{D}$, the QoM is a constant function of p for large p . These properties are all discussed in Section 3.3.4(v).

S-shaped utility

Fig. 3.15 presents the QoM results for the S-shaped utility function U_S . Although we do not have corresponding analytical results for S-shaped utility, note

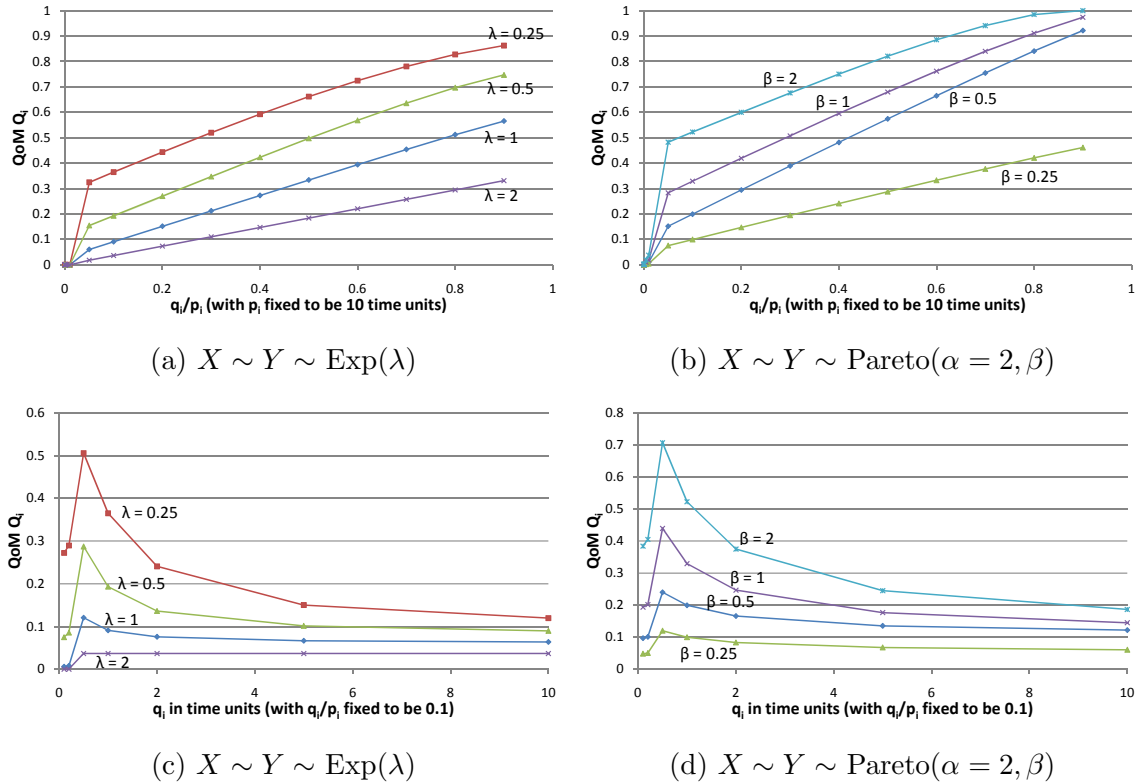


Figure 3.14. Achieved QoM for events that stay and have the delayed-step utility function U_D , $D = 0.5$.

from Fig. 3.15 that the results are similar to Delayed-step utility. This is due to the resemblance between the two utility functions.

Multiple sensor scenario

We now evaluate the multiple-sensor scenario discussed and analyzed in Section 3.3.5. We first summarize the simulation results and their interpretation, before presenting the detailed results.

1. The QoM is a monotonically increasing function of m . This is easily understood as the effective coverage time is equal to $m q$ which is proportional to the number of sensors present.
2. For *concave* utility functions, such as the step, exponential, and linear functions, the QoM decreases with p and r . This is because a larger p effectively leads

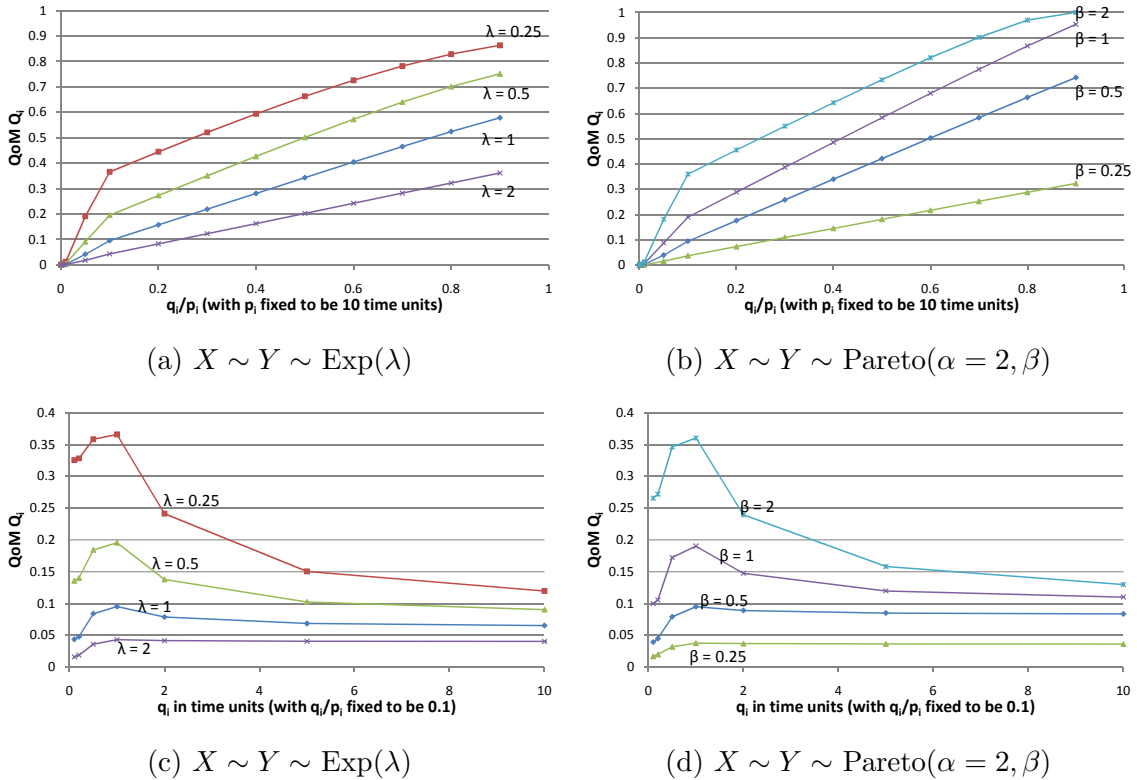


Figure 3.15. Achieved QoM for events that stay and have the S-shaped utility function U_S .

to a coarser fairness granularity. By the analysis in Section 3.3.5, for concave utility functions, a finer granularity gives a higher QoM. Similarly, a larger r means that the coverage times are more clustered together, i.e., $a \ll b$. This is equivalent to a coarser granularity leading to missed events and also redundancy in observing the same event.

- For non-concave utility functions such as the delayed-step and S-shaped ones, the QoM initially increases and then decreases with r . The explanation is similar to the analysis in Section 3.3.5. In order to gain a significant amount of information about a single event, enough observation time must be achieved. This requires larger values of r . After the critical time is passed, a higher degree of clustering leads to redundancy in collecting the information.

We now proceed to discuss the detailed results. These results are all supported by the analysis in Section 3.3.5. As mentioned there, we will emphasize the behavior of the QoM as a function of the parameters p , m , and r . We consider the cases when there are 2, 4, 5, 10, 20, and 40 mobile sensors. We measure the QoM Q_i achieved over 1,000,000 time units in a simulation run, and report the average Q_i of 20 different runs. The event staying and absent times are both Exponentially distributed with $\lambda = 1$, and the share $q/p = 0.01$.

Non-decreasing concave utility functions

We present results for the non-decreasing concave utility functions, namely, the step utility function U_I , exponential utility function U_E (with $A = 5$), and linear utility function U_L (with $M = 1$). Figs. 3.16, 3.17, and 3.18 show the achieved QoM as a function of the number of mobile sensors m and the clustering ratio r for the Step, Exponential, and Linear utility functions, respectively. The figures show that the achieved deployment QoM Q_i is the highest when the visits are evenly distributed, i.e., $r = 1$. As the visits are more clustered together, i.e., r is larger, Q_i is worsened when the interval of consecutive visits, if they are evenly distributed, is comparable to the event staying time. It is because when we have a concave utility function, it is generally more productive to monitor more events than to observe the same event for longer time. For Linear utility, however, notice from Fig. 3.18 that the separation of consecutive visits to a PoI has insignificant effects on the achieved deployment QoM. This is due to the nature of the linear utility function, i.e., it is indifferent to Q_i whether the same event or different events are monitored.

The figures also show that the achieved deployment QoM Q_i may grow sub-linearly with the number of sensors. It is because as the interval of consecutive visits, if they are evenly distributed in one period, is comparable or smaller than the event staying time, then it is more likely for the sensors to capture the same event, which results in no improvement in Q_i .

When the visit time of a sensor at the PoI is comparable to the event dynamics as depicted in Fig. 3.16(c), Q_i can only be slightly worsened, if it is affected at all, when

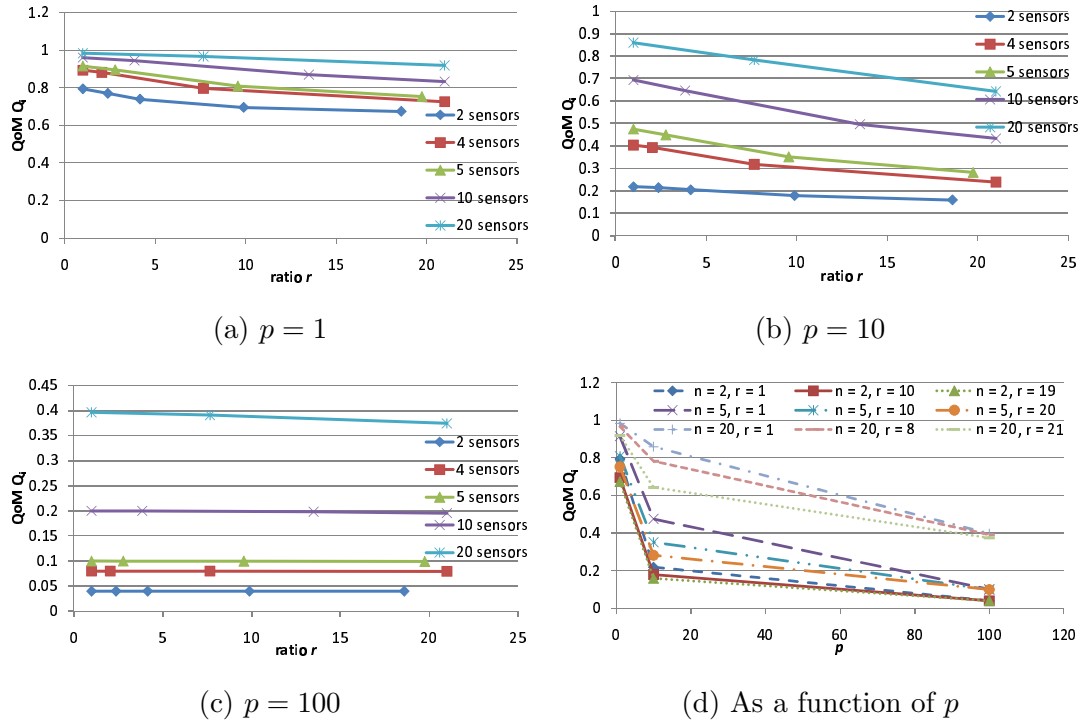


Figure 3.16. Achieved deployment QoM Q_i for staying events with step utility function U_I . $q/p = 0.01$, $X \sim Y \sim \text{Exp}(\lambda = 1)$.

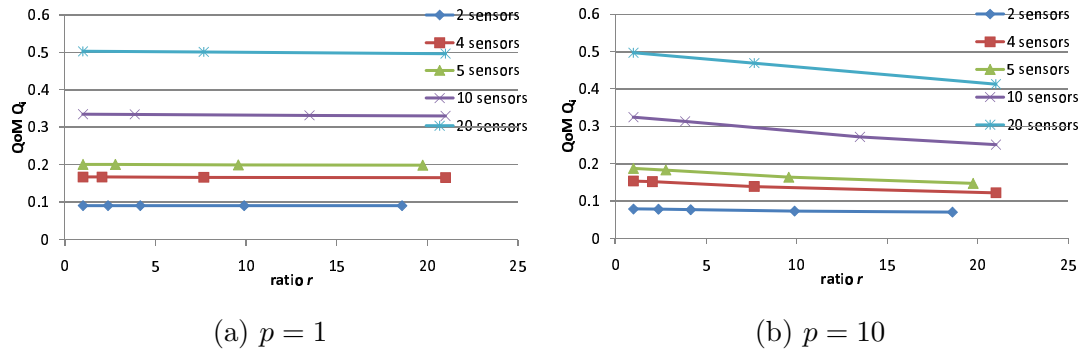


Figure 3.17. Achieved deployment QoM Q_i for staying events with exponential utility function U_E , $A = 5$. $q/p = 0.01$, $X \sim Y \sim \text{Exp}(\lambda = 1)$.

the sensors are more clustered together. It is because in such a situation, different visits to the PoI are likely to observe different events, and a more clustered group of sensors only reduces the probability of capturing new events.

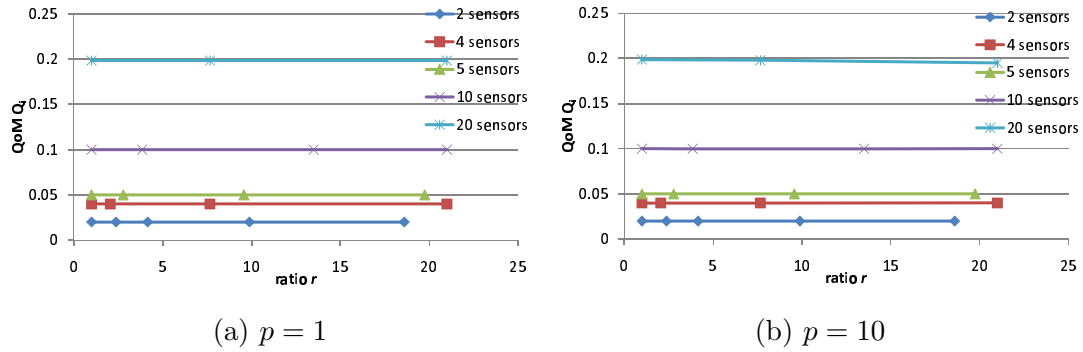


Figure 3.18. Achieved deployment QoM Q_i for staying events with linear utility function U_L , $M = 1$. $q/p = 0.01$, $X \sim Y \sim \text{Exp}(\lambda = 1)$.

Utility functions as a combination of convex and then concave parts

We now present results for the S-shaped utility function U_S and the delayed-step utility function U_D ($D = 0.5$ time units). Each can be seen as a combination of two different parts, a convex part followed by a concave part. Figs. 3.19 and 3.20 depict the achieved QoM as a function of the number of mobile sensors m and the clustering ratio r for the S-shaped and delayed-step functions, respectively. The figures show that clustering sensors together can improve the achieved deployment QoM as the separation of consecutive visits, when they are evenly distributed, is comparable to the event present time. It is because in such a scenario, clustering sensors together can help monitor the same event for longer in the convex part of the utility function, so that it is more likely to approach the information threshold for a better Q_i . However, Figs. 3.19(d) and 3.20(d) show that when the sensor visits are too clustered together, so that they collect information in the concave part of the utility function instead of capturing new events in the convex part of the function, the overall Q_i will suffer.

Fig. 3.20 also shows that the achieved deployment QoM Q_i may grow super-linearly with the number of sensors. It is because as the interval of consecutive visits, when they are evenly distributed in one period, is comparable or smaller than the event staying time, then it is more likely for the sensors to monitor the same event

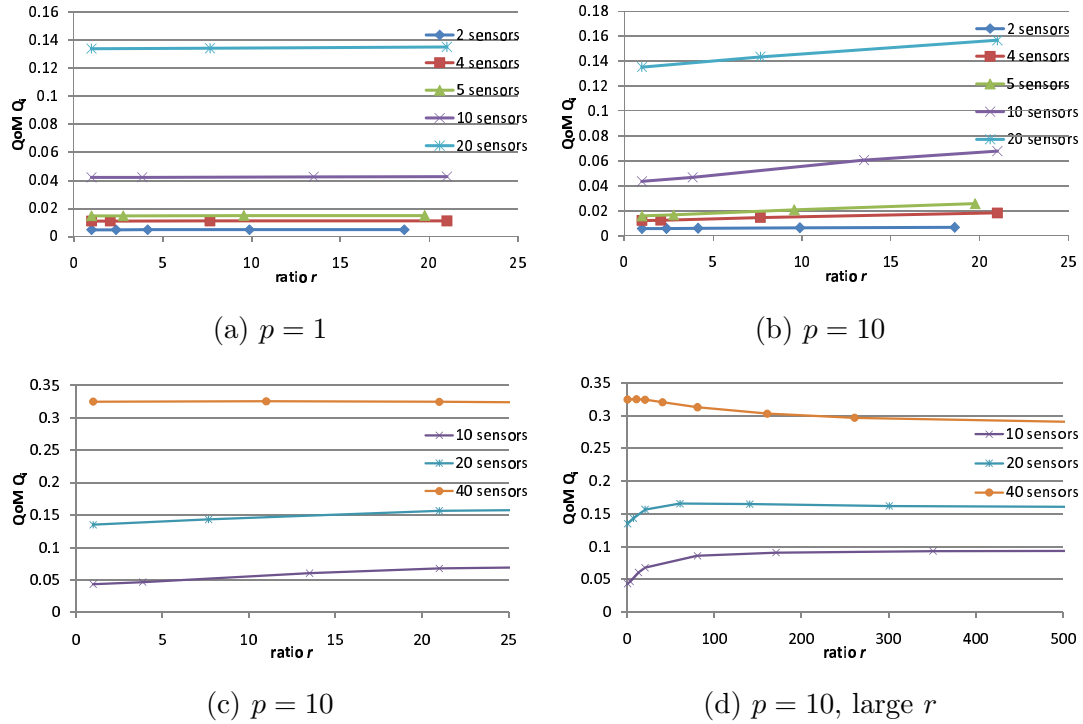


Figure 3.19. Achieved deployment QoM Q_i for staying events with S-shaped utility function U_S . $q/p = 0.01$, $X \sim Y \sim \text{Exp}(\lambda = 1)$.

and aggregate information within the threshold for productive information gain, which results in a sharp increase in the utility and an improvement in Q_i .

3.4.2 QoM of WRW-aLP

We now present the QoM performance of our proposed WRW-aLP mobility algorithm (Section 2.3.1). We only consider events of exponential dynamics.

(A) Single sensor using Residential Maps.

We evaluate the performance of the WRW-aLP algorithm and the BAI algorithm [25]. We show the results for real-life topologies, namely residential regions in Chicago, San Francisco, San Jose. The three city maps and their corresponding threat profiles are depicted in Fig. 3.21. The sensing utility function is the concave function given in Fig. 3.3. We use the dynamic events defined in Section 3.1.2. The

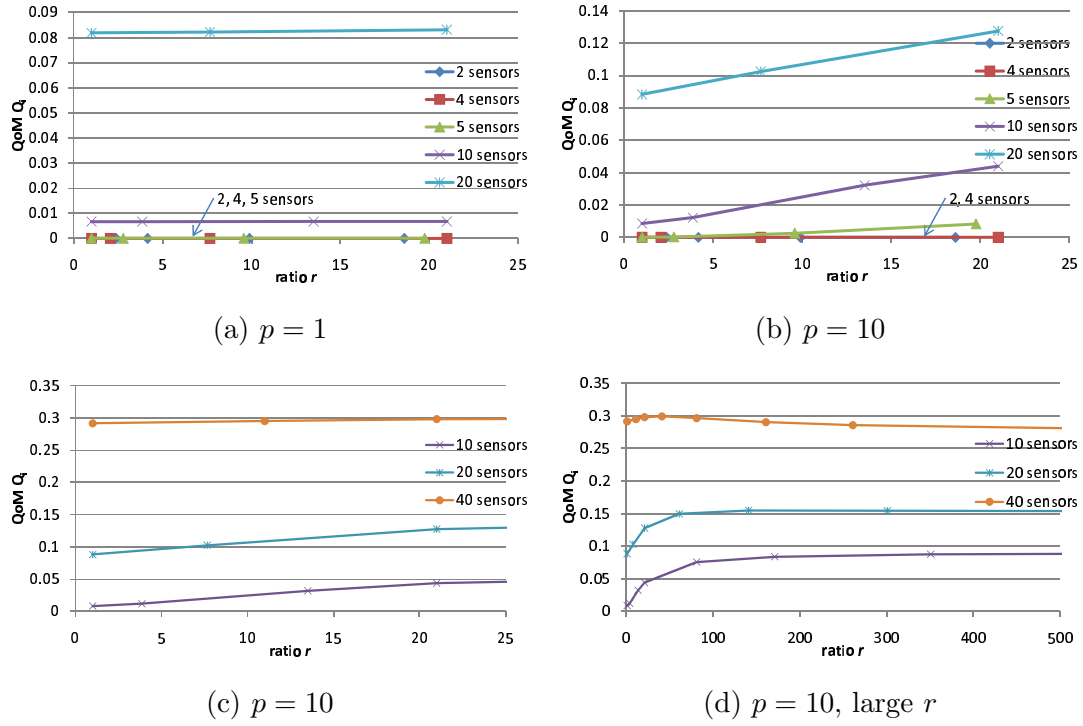


Figure 3.20. Achieved deployment QoM Q_i for staying events with delayed-step utility function U_D , $D = 0.5$ time units. $q/p = 0.01$, $X \sim Y \sim \text{Exp}(\lambda = 1)$.

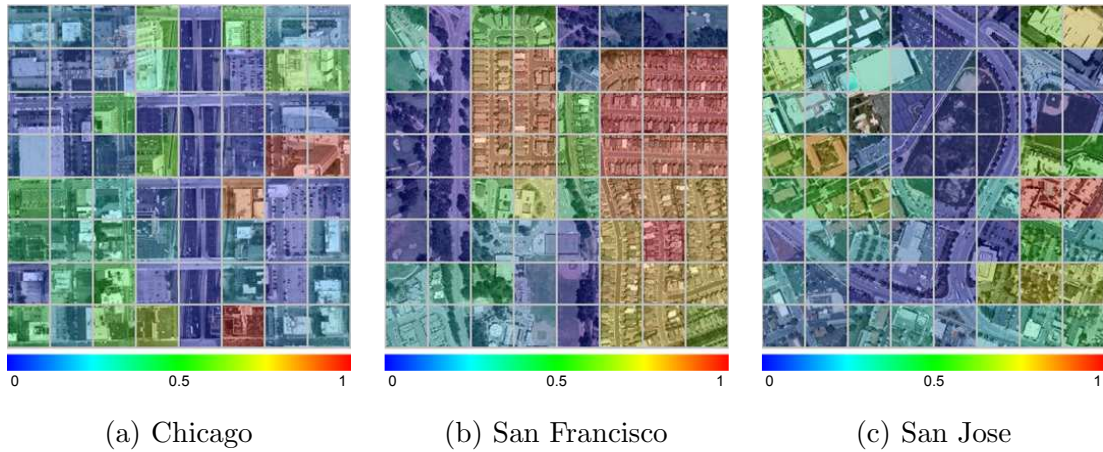


Figure 3.21. The three city maps and their corresponding threat profiles.

event durations are exponentially distributed with mean 13 minutes. Their absent times at a cell are exponentially distributed with mean value of 1 hour.

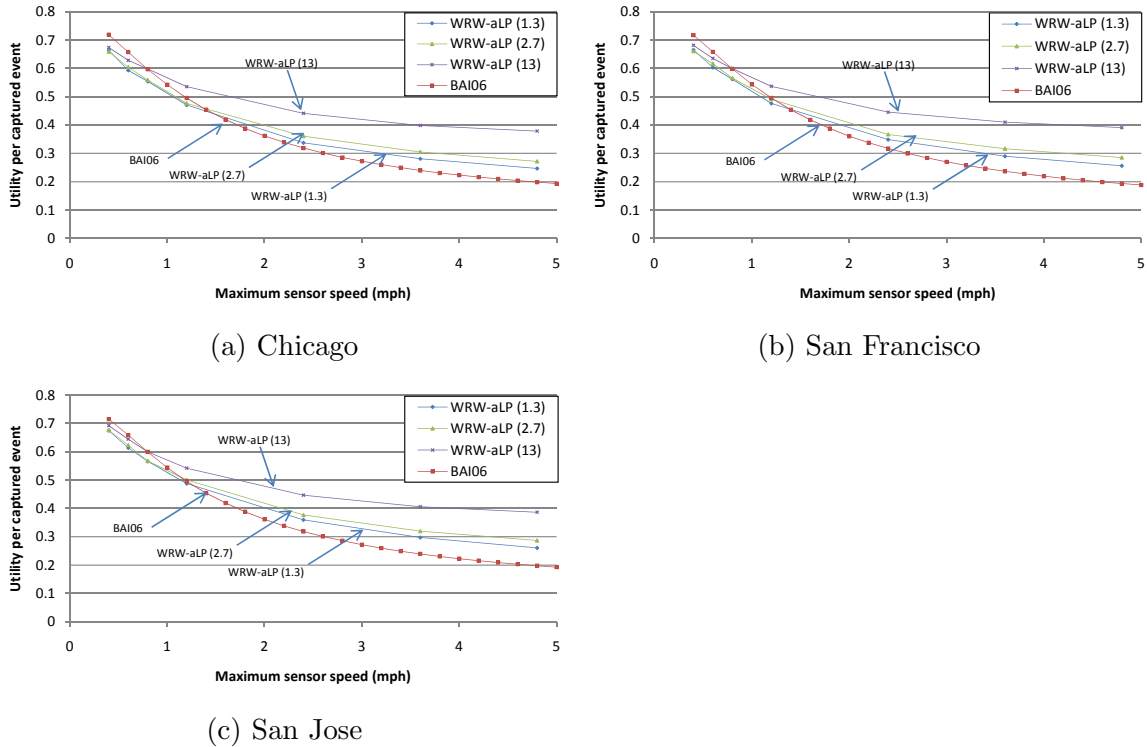


Figure 3.22. Per captured event utilities as a function of the maximum sensor speed.

Figs. 3.22 (a), (b), and (c) show the per-captured event utilities achieved for San Francisco, San Jose, and Chicago, respectively, by BAI and WRW-aLP (with varying P given in minutes) for different maximum speeds of the sensor. The results show that (1) pausing, which is possible with WRW-aLP and increases with P , can significantly increase the utilities of the sensing results; and (2) as the sensor speed increases, the utilities do not increase, but rather decrease consistently.

(B) Multi-sensors using Residential Maps.

In this section, we investigate the effects of multiple sensors. For simplicity, the sensors are assumed to coordinate through NC, i.e., they run independently in the surveillance area, under the same mobility algorithm.

Fig. 3.23(a) shows the normalized utility achieved by WRW-aLP(13) and BAI06 for 1, 2, and 4 sensors using the exponential utility function, U_E . In the figure, each

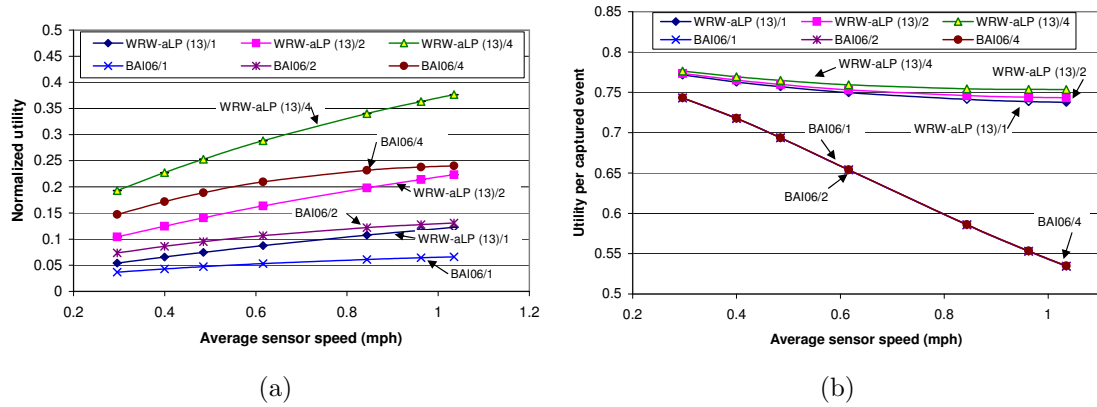


Figure 3.23. Performance of WRW-aLP and BAI with 1, 2, and 4 sensors: (a) normalized utility, and (b) utility per captured event.

plot is labeled “algorithm/number of sensors”; e.g., “BAI06/2” refers to the BAI06 algorithm with two sensors. For both algorithms, the normalized utility roughly doubles when the number of sensors doubles. The exact performance of n times more sensors improves by slightly less than a factor of n (from several to about 10% less). This is because the sensors work independently. Without coordination, it is possible for two sensors to cover the same PoI at the same time, and the redundant coverage causes an efficiency loss. However, the relative small efficiency loss in these experiments shows that the redundancy is not severe when 10 PoIs are served by up to four sensors.

In terms of the utility per captured event, shown in Fig. 3.23(b), the improvement is quite small with more sensors. For example, the per-captured event utilities of WRW-aLP(13) are all about 0.76 whether 1, 2, or 4 sensors are used. The results show that the higher normalized utility achieved by more sensors is mostly due to proportionately more events captured by the additional sensors.

(C) Simulation Results (Ring Topology)

In this section, we discuss results for the ring topology in Fig. 3.24. The same topology is used in [25] to evaluate the BAI algorithm. The ring consists of a sequence of 50 cells, each of dimension 250 ft. \times 250 ft., and 10 PoIs are uniformly placed on

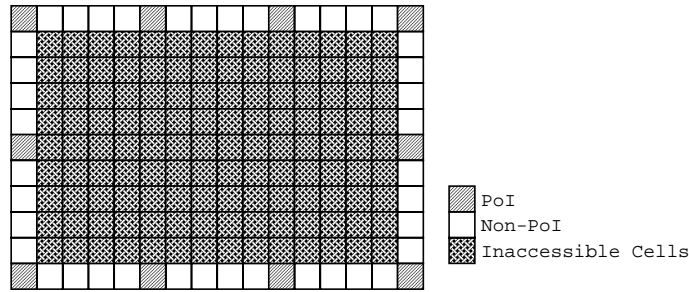


Figure 3.24. The ring topology.

the ring. We use the same sensing utility function and the same type of dynamic events as in Subsection 3.4.2(A). The event staying and absent times are exponentially distributed with mean 800 s. For WRW-aLP, L is set to be 1500 ft. The pause time parameter P is varied to control the amount of sensor movement. We use WRW-aLP(k) to denote the algorithm running with $P = k$ minutes.

To assess the impact of mobility, we also compare with *best-case* static coverage, in which each sensor's static position is chosen to give the best performance. We report results that are averages of at least 10 different runs. The standard deviations are very small (within 1% of the averages). *Hence, we do not report the standard deviations or the error bars.*

In this set of experiments, one sensor is used to cover the whole area. This corresponds to a severely constrained resource environment, in which the resource availability is only 10% (i.e., one sensor for 10 PoIs).

(1) Normalized Utility. We compare WRW-aLP (with different values of P), BAI, and static coverage by the normalized utility measure defined as follows,

- **Normalized utility of the events captured.** This is the sum of utilities of all the events captured within a given time interval, normalized by the total number of events that appear during the time interval. A higher normalized utility shows that the sensors can collect a larger total amount or fraction of the interesting information.

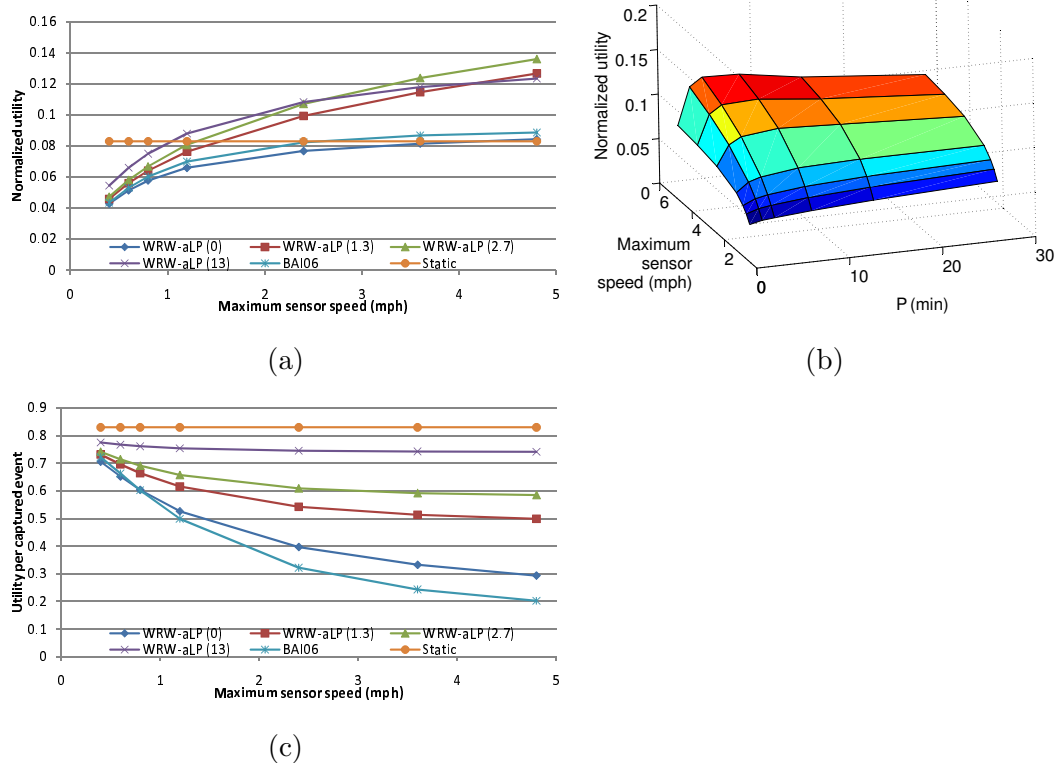


Figure 3.25. Normalized utility as a function of (a) maximum sensor speed, (b) different WRW-aLP parameters. (c) Utility per-captured event as a function of maximum sensor speed.

Fig. 3.25(a) plots the normalized utility achieved by the different algorithms as a function of the sensor's maximum speed. The following observations are in order:

(i) For static coverage, the sensor always stays at one PoI. Hence, it should be able to capture 10% of the events at their maximum utilities. From Fig. 3.25(a), however, notice that static coverage has a normalized utility of about 0.08, which is less than 0.1. This is because some of the events are short-lived, and do not last long enough for them to be captured at utility one.

(ii) From Fig. 3.25(a), notice that WRW-aLP(0) has similar performance as BAI. This is because $P = 0$ ensures that the sensor will continuously move between the PoIs, similar to the BAI algorithm. When P increases to 80 seconds, however, WRW-aLP(1.3) can perform significantly better than BAI. The results show that pausing

at PoIs can improve the quality of the sensing by allowing the events to be measured for longer and therefore with higher confidence.

(iii) Static coverage is extremely efficient. Hence, while it is inherently unfair, it might perform the best purely from a utility standpoint. Fig. 3.25(a), however, shows that WRW-aLP always outperforms static coverage when the maximum speed exceeds a modest value. This is partly due to the concavity of the utility function. When the utility function is concave, much of the utility is obtained during the initial period of observing a new event. This encourages the sensor to occasionally move from one PoI to another in order to catch more new events, as long as the moving speed is not too low to make the travel overhead too high.

The normalized utility as a function of P and the maximum sensor speed is shown in Fig. 3.25(b). Notice that the binary function is concave in both arguments, showing that standard techniques can be applied to compute, for example, the optimal P for maximizing performance.

(2) Utility per captured event. Fig. 3.25(c) compares the utility per captured event for the different algorithms as a function of the maximum sensor speed. Notice from Fig. 3.25(c) that WRW-aLP(p), for $p \geq 1$, achieves a significantly higher per-captured event utility than BAI or WRW-aLP(0) for the same maximum speed. Hence, whereas the previous results show that a positive pause time with WRW-aLP achieves a higher normalized utility than a continuous movement algorithm such as BAI or WRW-aLP(0), these results further show that they do so not by capturing more events, but by improving the confidence of each captured event. For the continuous movement algorithms, notice also that the per-captured event utility drops significantly as the sensor speed increases. This shows that for sensing with a temporal dimension, too much mobility can be counterproductive.

(3) Performance of other utility functions. We further present the performance of the mobility algorithms when the utility of event sensing is given by other utility functions given in Fig. 3.2.

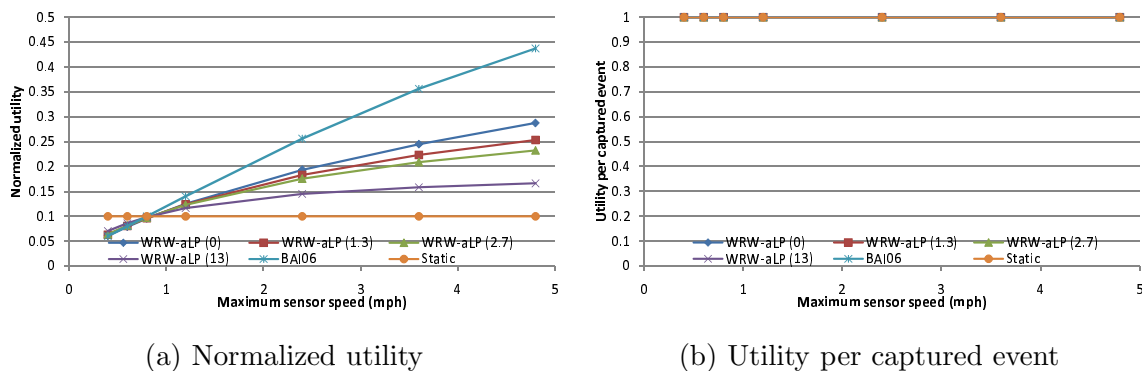


Figure 3.26. Achieved utility as a function of maximum sensor speed using step utility function U_I .

(i) *Step utility.* Fig. 3.26 depicts the results for step utility function. The figure shows that pausing hurts performance in normalized utility.

It is because as utility is achieved instantaneously when the event is captured, it is beneficial to move on and capture new events at other PoIs instead of waiting for the next event to appear at the same PoI.

(ii) *Linear utility.* Fig. 3.27 depicts the results for linear utility function. The figure shows that a longer pause time is beneficial to achieve both a higher normalized utility and utility per captured event. The larger the pause time, the better the normalized utility. It is because by using Linear utility, it does not matter if the sensor is capturing the same event or a new event, provided that the maximum utility has not yet reached. By having a longer pause time, the mobile sensor spends less time on traveling over non-PoIs. At the same time, by traveling occasionally, the mobile sensor can capture new events at different PoI instead of waiting for the next event or capturing the same event beyond the time to reach maximum utility at the same PoI.

(iii) *Delayed-step utility.* Fig. 3.28 depicts the results for step utility function. The figure shows that pausing is beneficial to achieve better performance in normalized utility and utility per captured event. It is because a threshold amount of monitoring time for an event is necessary to achieve non-zero utility under the

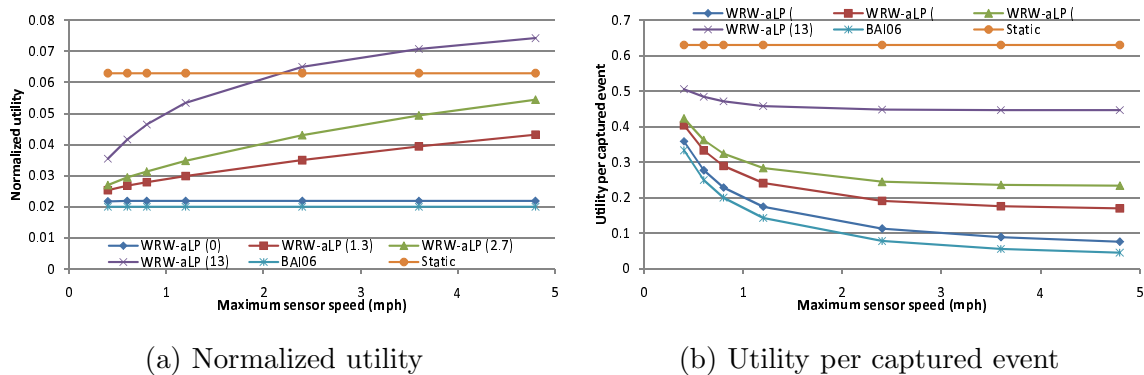


Figure 3.27. Achieved utility as a function of maximum sensor speed using linear utility function U_L .

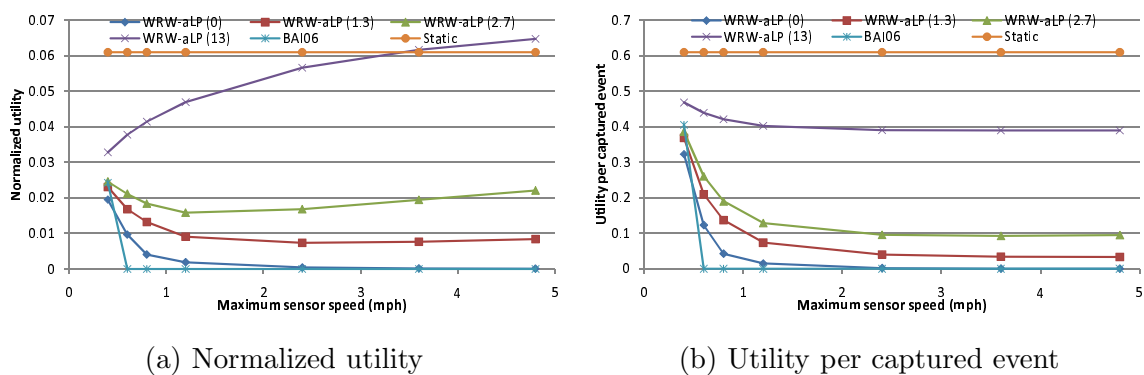


Figure 3.28. Achieved utility as a function of maximum sensor speed using delayed-step utility function U_D .

delayed-step utility function. Hence, if the pause time of the mobile sensor is not long enough, it can hardly achieve that threshold monitoring time.

For WRW-aLP with small pause time, an initial increase in speed hurts normalized performance because the drop in utility per captured event under delayed-step utility function is significant when speed is initially increased, while the number of new events being captured with non-zero utility is not increased fast enough to compensate for the lose of utility in each monitored event.

(iv) *S-shaped utility.* Fig. 3.29 depicts the results for s-shaped utility function. The figure shows similar performance as delayed-step utility function, but the achieved

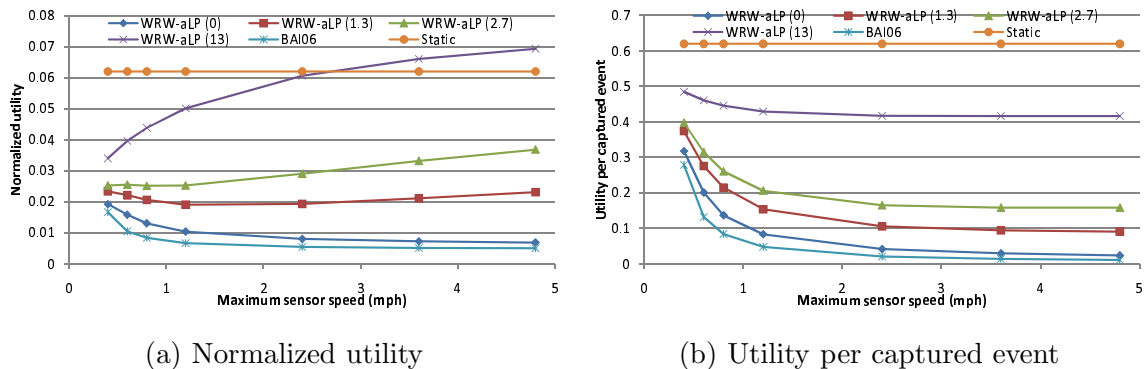


Figure 3.29. Achieved utility as a function of the maximum sensor speed using S-shaped utility function U_S .

performance is better. it is because the drop of utility per captured event for mobile sensors is less significant when speed is increased initially, which is due to the more gradual change in utility as monitoring time is reduced than the delayed-step utility function.

3.5 Summary

In this chapter we have presented extensive analysis to understand the QoM properties of periodic proportional-share mobile sensor coverage. We show that (1) A higher share of the coverage time generally increases the QoM, but the relationship is not linear. (2) For staying events, the QoM can be much higher than the proportional share, due to the observation of “extra” events that arrive when the sensor is not present. This justifies mobile coverage from an information-capture point of view, i.e., the sensor gains by moving between places to search for new information. (3) The event utility function is important in determining the optimal fairness granularity p . For concave utility functions such as Step, Exponential, and Linear utilities, the QoM monotonically decreases with p , whereas for Delayed-step and S-Shaped utilities, the QoM generally peaks at an intermediate p . Our analysis for Exponential/Pareto event

dynamics and different forms of the utility function is all supported by the simulation results.

We have also analyzed the costs/benefits of a stateful stochastic mobile coverage algorithm for sensing tasks with a temporal dimension. Our results show that when sensor resources are constrained, a limited amount of sensor movement can provide threat-based fairness. Moreover, the quality of the sensing can be even better than best-case static coverage. Comparisons between WRW-aLP and BAI have provided insights on how mobility may impact our performance differently than simple event capture. We show that, in contrast to existing basic results on simple event capture, too much mobility may be counterproductive in our case, by reducing the accuracy of threat-based coverage and compromising the utility of the sensing results.

4 PRIVACY STUDY

More economical but less precise sensors can be attached to the handheld devices used by the general public, especially with the advances in the sensor technologies in recent years, which have made the sensors smaller, lighter, cheaper, and more power efficient. To facilitate the design and evaluate the performance of using the general public as the backbone of a mobile sensor network, mobility traces of different entities are collected and published. In this chapter, we study the privacy vulnerability of published anonymous trace sets with different characteristics when the location of the participants can be openly observed or learned by an adversary who tries to identify the complete path history of the participant(s) [41].

4.1 System Model

We assume a set of traces recording the movement of some mobile nodes over a period of time is published and made available to everyone. Participants are mobile nodes that have their movement being published in the trace set. The traces may be snapshots or samplings of the movement of the participants. To protect the privacy of the participants, the ID of the traces are consistently replaced with a random ID which is not correlated with the real ID. The granularity in the spatial and temporal domains may also be reduced for privacy protection. In particular, the location of a participant may be reported as inside an area of specific size, which we called a *cell*, instead of being reported at a specific location precisely. There is an adversary who tries to identify the complete trace of one or more participants from the published trace set. The adversary does so by collecting snapshots of the movement of the participant(s), which we named as *side information*, from sources other than the published traces, and uses appropriate strategies to combine the two information

sources and identify the complete trace of the participant. The participant(s) the adversary tries to identify is/are named as victim(s).

Notations to be used in this chapter are summarized as follows,

- Θ : The collection of all cell IDs.
- $\{L_i\}_{i=1,2,\dots,N}$: The collection of all the traces of the participants, each indexed by an anonymous index i . N is the total number of traces. More specifically, for each i , L_i is a function of time $L_i : \mathbf{R}_+ \longrightarrow \Theta$ giving the ID of the cell visited by participant i .
- $\{s_k\}_{k=1,2,\dots}$: The sampled times at which the actual node locations are published, i.e., $L_i(s_k)$ is the published ID of the cell of the mobile node i at time s_k .
- R : The noisy side information of the victim.
- $\{t_k\}_{k=1,2,\dots}$: The time instants corresponding to the victim's locations in the sequence of noisy pieces of side information about the victim, i.e., $R(t_k)$ is the (perturbed) ID of the cell visited by the victim at time t_k as revealed to the adversary.
- P_{ij} : The transition probability for a node to go from cell i to cell j in some pre-defined time interval T .

In order to concentrate on the key issue of privacy breach, we further make the following assumptions:

1. The sampled times s_k 's are equally spaced.
2. The noise in the side information in each revelation instance is assumed to be some iid random variable Z_k 's of some given distribution \Pr_Z . Hence we have

$$R(t_k) = L_{i^*}(t_k) + Z_k, \quad (4.1)$$

where i^* is the victim's trace ID (which is of course not known to the adversary).

3. All the mobile nodes are assumed to have the same movement model which is assumed to be Markovian. Hence the statistics of the whole collection of traces can be completely described by one single transition matrix P_{ij} . This matrix is either given or estimated by some general world knowledge.

4.2 Related Work

Various privacy issues of published data sets have been studied in the literature [42–49]. Sweeney [42] proposes a protection model named *k-anonymity* and a set of accompanying policies for privacy protection. When *k-anonymity* is satisfied, each individual is indistinguishable from $k - 1$ other individuals. Bayardo and Agrawal [43] propose a practical method to identify a provably optimal *k-anonymization* of real census data, or a good anonymization when the optimal one cannot be found in reasonable time, because the general problem is NP-hard. Xiao and Tao [44] propose a generalization principle of *m-invariance* to effectively limit the risk of privacy disclosure in data re-publications, given the many potential correlations among various snapshots of each data entry in subsequent publications that can be used to derive sensitive information. Martin *et al.* [45] quantify the impact of background knowledge possessed by an attacker on privacy breach. They express the attacker’s background knowledge in a language, and provide an algorithm to determine the amount of disclosed sensitive information in the worst case with respect to the amount of the background knowledge. Nergiz *et al.* [46] extend the notation of *k-anonymity* for trajectories and propose a generalization-based approach to publish trajectories for further privacy protection. Abul *et al.* [47] propose the use of space translation to achieve (k, δ) -anonymity for moving objects databases, where δ is the radius of a cylindrical volume representing the trajectory imprecision. In the data mining context, Agrawal and Srikant [48] propose a reconstruction procedure to build a decision-tree classifier without accessing the precise information in individual data records, so that the distributions of the data values can be reconstructed with sufficient ac-

curacy. They also suggest the use of value-class membership and value distortion to preserve privacy. [49] gives a detailed study on the state-of-the-art approaches to protect privacy while mining knowledge from databases of trajectories. The authors discuss technical and regulation approaches for privacy protection on trace collection, processing, access, and publication, and knowledge discovery and sharing.

Narayanan and Shmatikov [50] study the privacy implications of releasing anonymous and perturbed user ratings of movies in the Netflix database for a research competition to better predict customers' preferences for different new movies, such that the revenue of Netflix can be improved. They propose an effective algorithm to de-anonymize the data set, and verify its performance with the published data. Their problem, in which an adversary is given perturbed side information about a victim to identify the victim's record among all the database records, is one of the attack scenarios we consider in this paper.

Privacy protection of mobile nodes in location-based services has also been studied [51–54]. One proposed approach is to reduce the spatial/temporal granularity of the location information made available to the service provider while achieving satisfactory service effectiveness [51, 52]. Hoh *et al.* [53] devise a protection strategy to release user data only when certain privacy constraints are met. Meyerowitz and Choudhury [54] suggest sending fake requests with the real ones to reduce the ability for an adversary to trace a mobile node over time. A summary of the above approaches can also be found in [49].

Our problem differs from the previous work by its specific focus on privacy leaks of user location information considering the characteristics of anonymous mobility traces of users, under the assumption that an adversary, assisted by different amounts of side information that can be realistically obtained, employs various well grounded strategies to infer the private information. As discussed in Chapter 1, we are motivated by the emerging practice of collecting traces of real users in a mobile network and publishing anonymized and possibly cloaked versions of these traces through various data portals, to assist in the design and evaluation of these networks.

4.3 Adversary Strategies

In this section, we present the attack strategies to be used by the adversary.

4.3.1 Passive Adversary

There are many ways an adversary can collect the snapshots of a victim without physically observing the victim. For instance, snapshots may be collected through press releases, news, web blogs, online social networks, tweets, etc. After collecting such snapshots, the adversary may need to infer the location of the victim at the sampling time of the published traces. The adversary can do so by consulting a transition matrix which describes the general movement pattern of nodes in the network area, which may be resulted from the nature of the nodes, physical constraints, and speed limit. The adversary then applies Bayesian Inference or use the maximum likelihood estimator (MLE) to identify the trace that gives the best match with these pieces of side information. Notice that because the adversary does not personally observe the victims, precision of the side information is compromised, and errors may be assumed.

The goal of the attack strategy is that given R , find the L_i that gives the best match. The formulation of such a procedure is described below. Given $\{R(t_k)_{k=1,2,\dots}\}$, compute

$$\begin{aligned} & \Pr(L_i | \{R(t_k), k = 1, 2, \dots\}) \\ &= \frac{\Pr(L_i, R(t_k), k = 1, 2, \dots)}{\Pr(R(t_k), k = 1, 2, \dots)} \\ &= \frac{\Pr(R(t_k), k = 1, 2, \dots | L_i) \Pr(L_i)}{\sum_{j=1}^N \Pr(R(t_k), k = 1, 2, \dots | L_j) \Pr(L_j)}. \end{aligned} \quad (4.2)$$

The goal of the MLE is to find i which maximizes the expression (4.2). Note that the denominator is a constant. In addition, without any knowledge about how the victim is chosen, we set the a priori distribution of the victim to be uniform: $P(L_i) = \frac{1}{N}$ for $i = 1, 2, \dots, N$. Hence the solution of the MLE is given by:

$$\max_{i=1,2,\dots,N} \Pr(R(t_k), k = 1, 2, \dots | L_i). \quad (4.3)$$

With the assumption of the noise model given in (4.1), the expression (4.3) can be given in the following form:

- **Side information references time instants that coincide with sampled times in the trace only, case A1.** Because the noise is iid, we have

$$\Pr(R(t_k), k = 1, 2, \dots | L_i) = \prod_k \Pr_Z(R(t_k) - L_i(t_k)). \quad (4.4)$$

The assumption of equally spaced sampled times s_k 's means $\{t_k : k = 1, 2, \dots\} = \{s_k : k = 1, 2, \dots\}$; then $T = s_2 - s_1$.

- **Side information may reference time instants between two consecutive sampled times in the set of traces, case A2.** By the Markovian assumption of the movement model, (4.3) can be given by:

$$\begin{aligned} & \Pr(R(t_k), k = 1, 2, \dots | L_i) \\ &= \Pr(R(t_k), k = 1, 2, \dots | L_i(s_j), j = 1, 2, \dots) \\ &= \frac{\prod_k \left[\Pr(L_i(s_{\tilde{k}+1}) | R(t_k)) \times \Pr(R(t_k) | L_i(s_{\tilde{k}})) \right]}{\prod_k \left[\Pr(L_i(s_{\tilde{k}+1}) | L_i(s_{\tilde{k}})) \right]}, \end{aligned} \quad (4.5)$$

which can be easily expressed in terms of the transition matrix P_{ij} .

The assumption of equally spaced sampled times s_k 's means $\{t_k : k = 1, 2, \dots\} = \{s_k : k = 1, 2, \dots\}$; then we assume that for each t_k , there exists \tilde{k} such that $s_{\tilde{k}} < t_k < s_{\tilde{k}+1}$ and $t_k = \frac{1}{2}(s_{\tilde{k}} + s_{\tilde{k}+1})$. In this case, we take $T = \frac{1}{2}(s_2 - s_1)$.

Note that the expression (4.4) takes on the particularly simple form if the noise Z_k 's are iid *Gaussian random variables* $N(0, \sigma^2)$:

$$\Pr(R(t_k), k = 1, 2, \dots | L_i) = C \exp -\frac{1}{2\sigma^2} \sum_k |R(t_k) - L_i(t_k)|^2, \quad (4.6)$$

for some constant C . Hence, the MLE is the same as the following *minimum square* approach:

$$\min_i \sum_k \left| R(t_k) - L_i(t_k) \right|^2. \quad (4.7)$$

The above provides a rigorous mathematical formulation for the Bayesian inferencing equipped with the side information. On the other hand, the above also leads to some simplified heuristic approaches for tackling the victim identification problem. Qualitatively, they are all similar to the minimum square approach. We find it a useful contribution to record and compare their performance. In the following we consider four strategies used by the adversary to identify the victim's trace from the published trace set, when it collects side information $\{R(t_k) : k = 1, 2, \dots, M\}$ of the victim node from third parties. We first describe them for case **A1**:

1. **MLE Approach (MLE)**. This is the same as formulation (4.4), i.e., the *similarity* value of trace i is given by $\prod_k \Pr_Z(R(t_k) - L_i(t_k))$. The trace with the *maximum similarity value* is declared to be the victim's.
2. **Minimum Square Approach (MSQ)**. This is the same as formulation (4.7), i.e., the *similarity* value of trace i is given by $-\sum_k \left| R(t_k) - L_i(t_k) \right|^2$. The trace with the *least negative similarity value* is declared to be the victim's.
3. **Basic Approach (BAS)**. In this approach, the adversary assumes that the noise is zero-mean and has a specific standard deviation (σ), but makes no assumption about its exact distribution. The adversary then computes the *similarity* value of trace i with the collected side information using the following equation:

$$\sum_{k=1}^M I(R(t_k), L_i(t_k)), \quad (4.8)$$

where $I(x, y) = 1$ if $|x - y| \leq 2\sigma$ and 0 otherwise. Hence, the adversary accepts a trace as a potential candidate if it is possible for the trace owner to appear in a radius of $2 \times \sigma$ of the revealed location, which encloses all possible noise if it is uniformly distributed, or 95.6% of noise if it is Gaussian. The trace with the *maximum similarity value* is declared to be the victim's.

4. **Weighted Exponential Approach (EXP)**. In this approach, which is proposed and analyzed in [50], we assume that the adversary does not know the

type of noise or its magnitude. Similar to **BAS**, the adversary computes and maximizes the *similarity* value of trace i using the following equation,

$$\sum_{k=1}^M \frac{1}{\text{Weight}(R(t_k))} \exp \left\{ -\frac{1}{C} |R(t_k) - L_i(t_k)| \right\}, \quad (4.9)$$

where $\text{Weight}(R(t_k))$ is some weight assigned to the revealed cell $R(t_k)$ and C is a constant. In the simulations we assign equal weights to all of the cells because with possible errors in the revealed location, it is unclear how different weights could be assigned.

The above formula can be easily modified for case **A2**. For convenience, we first define for each trace i , the function $P_i : \Theta \times \{t_k : k = 1, 2, \dots, M\} \rightarrow \mathbf{R}_+$:

$$P_i(l, t_k) = \frac{P_{x,l} P_{l,y}}{P_{x,y}},$$

where $x = L_i(s_{\bar{k}})$, $y = L_i(s_{\bar{k}+1})$, and $s_{\bar{k}} < t_k < s_{\bar{k}+1}$. Then we have,

- **MLE₂**:

$$\Pi_k \left(\sum_{l \in \Theta} P_i(l, t_k) \Pr_Z(R(t_k) - l) \right). \quad (4.42)$$

- **MSQ₂**:

$$\sum_k \left(\sum_{l \in \Theta} P_i(l, t_k) |R(t_k) - l|^2 \right). \quad (4.72)$$

- **BAS₂**:

$$\sum_{k=1}^M \left(\sum_{l \in \Theta} P_i(l, t_k) \times I(R(t_k), l) \right). \quad (4.82)$$

- **EXP₂**:

$$\sum_{k=1}^M \left(\sum_{l \in \Theta} \frac{P_i(l, t_k)}{\text{Weight}(R(t_k))} \exp \left\{ -\frac{1}{C} |R(t_k) - l| \right\} \right). \quad (4.92)$$

Notice that the four approaches have the same computational complexity, which is linear in the number of pieces of revealed side information and the number of nodes.

4.3.2 Active Adversary

The adversary can also personally observe the participants, remember the time and location it meets with a participant, and use these pieces of side information to infer the trace that gives the best match with each encountered participant. Fig. 4.1 details the strategy used by the adversary to identify the traces of the participants. The **Attack** program takes as input the traces that are published progressively. The algorithm first assumes that all the traces are candidate traces for each participant. A trace is said to be a candidate trace of a participant if it appears at the same set of times and locations as when/where the adversary meets the participant, and the trace has not been identified. As time evolves, the adversary removes candidate traces which do not agree with the observed information about each victim from the set for that victim. The function **Cascade** takes two input parameters, where **candidate_set** is the candidate set of all victim nodes and i is the victim ID identified. The function is called when a victim's trace is identified, so as to remove that trace from the candidate set of other victims.

Notice that the adversary may not identify a participant at times they meet each other, but the identification can occur at a later time when all but one of the candidate traces are identified and removed, as indicated by the recursive **Cascade** function call in Fig. 4.1. Hence, the adversary identifies a participant more efficiently when it tries to identify as many participants as possible in this scenario.

4.4 Evaluation

In this section we give a comprehensive evaluation of the privacy vulnerability problem in trace set publication. We first present the characteristics of the mobility traces we used in our evaluation. We then present the performance of the different privacy attack strategies of the adversary and explain the observations using the characteristics of the traces.

```

1  Cascade(candidate_set, i)
2    let j = trace id where candidate_seti = {j}
3    /* remove the identified trace from candidate set of other victims */
4    For (m = 0; m < number_of_trace; m++)
5      If trace j in candidate_setm and m = i
6        remove trace j from candidate_setm
7        If |candidate_setm| = 1
8          Cascade(candidate_set, m)
9        Endif
10     Endif
11   Endfor
12 Attack({Li}i=1,2,...N)
13 /* initially all traces are possible candidates to each victim */
14 For (m = 0; m < number_of_trace; m++)
15   add all traces to candidate_setm
16 Endfor
17 While (sampling_time not ended)
18   For each node i met at sampling_time and each trace j in candidate_seti
19     /* check if a candidate trace appear at the observed location */
20     If (met node i at location r at sampling_time and Lj(sampling_time) != r)
21       remove trace j from candidate_seti
22       If |candidate_seti| = 1
23         Cascade(candidate_set, i)
24       Endif
25     Endif
26   Endfor
27   report all identified victims in the current sampling_time
28   evolve sampling_time
29 Endwhile

```

Figure 4.1. Specification of active adversary attack algorithm.

4.4.1 Characteristics of Mobility Traces

We begin the evaluation by analyzing the differences in behaviors between the real traces and the simple synthetic traces. Their fundamental differences will be illustrated by six sets of mobility traces. They include two sets of real traces: (1) taxi cabs in the San Francisco area [55], and (2) buses in the ShangHai Grid system [10]. Basic statistics about these two sets of traces are listed in Table 4.1. We then make use of statistics from the San Francisco cab trace to generate four other synthetic traces using the random waypoint mobility model (*rway*) and two of its variants which impose a maximum trip length of x (in km) (*rway* - x with $x = 10, 20$), and the random walk model (*rwalk*). In particular, we use the number of cells visited by the cabs as the size of the map of the synthetic traces (approximated by a square of size $0.63^\circ \times 0.63^\circ$), the average speed of the cabs (about 13.8 mph) as that of the synthetic mobile nodes, and the average time the cabs are active (about 15 days) as the simulation run time of the synthetic traces. We assume that published traces are snapshots taken every minute with spatial granularity of 0.01° in latitude and longitude unless stated otherwise. Characteristics of traces are studied using the following four metrics. Observations that can be explained using differences between movement preferences of the mobile nodes are summarized at the end of this section. They are useful in establishing intuition about the attack strategies.

Correlation between traces

We use the Pearson product-moment correlation coefficient [56] to quantify the correlations between node pairs. For any mobile node pair i and j , the quantity is defined as follows.

$$C(i, j) = \lim_{M \rightarrow \infty} \frac{1}{M} \sum_{k=1}^M \frac{L_i(s_k) - EL_i}{\sigma_{L_i}} \frac{L_j(s_k) - EL_j}{\sigma_{L_j}},$$

Table 4.1
Basic statistics of the real traces.

	San Francisco cabs	ShangHai Grid buses
Min. latitude	37.05	30.7217
Max. latitude	38.00	31.5899
Min. longitude	-122.86	121.0001
Max. longitude	-122.00	121.9117
# cells ^a	8170	8004
# active cells ^b	3997	2108
# nodes	536	2348
Min. timestamp	Sat May 17	Mon Feb 19
(local time)	03:00:04 2008	08:00:01 2007
Max. timestamp	Tue June 10	Sat Feb 24
(local time)	02:25:34 2008	08:00:00 2007

^awhen spatial granularity is 0.01°.

^bcells ever visited by any node.

where EL_i and σ_{L_i} are respectively the average and standard deviation of node i 's locations:

$$EL_i = \lim_{M \rightarrow \infty} \frac{1}{M} \sum_{k=1}^M L_i(s_k),$$

$$\sigma_{L_i} = \lim_{M \rightarrow \infty} \frac{1}{M} \sum_{k=1}^M (L_i(s_k) - EL_i)^2.$$

The distribution of the correlations between different node pairs is depicted in Fig. 4.2.

The figure shows that movements of different cabs have little or no correlation, while those of the random walk nodes have higher correlations. It is because cabs are unlikely to follow each other for a long time. Random walk nodes show the highest correlation since their movements are synchronized and their choices of next movement are limited to the immediate neighboring cells. Although a low correlation

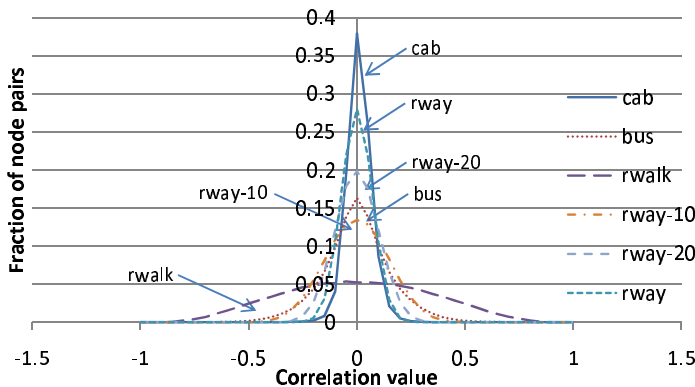


Figure 4.2. Distribution of correlations between traces of the same set.

between the traces indicates that they do not share common paths over a long time, having a high correlation between the traces does not imply otherwise. It is because the computed correlation is based on the distance relative to the mean position of each trace but not to a common location, and it does not take into account the orientation of the nodes.

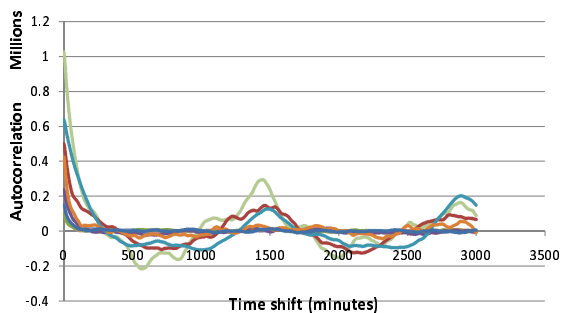
Autocorrelation of the same trace

The autocorrelation $C(i, s)$ of trace i with time shifting of s is defined as:

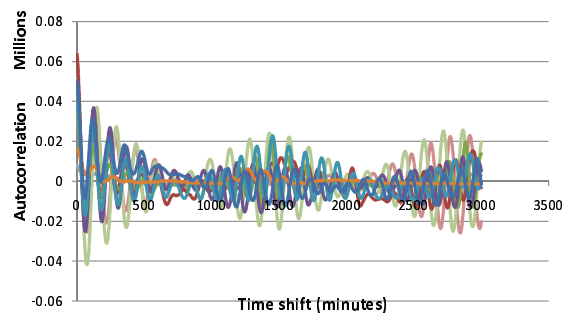
$$\lim_{M \rightarrow \infty} \frac{1}{M} \sum_{k=1}^M (L_i(s_k + s) - EL_i)(L_i(s_k) - EL_i).$$

Fig. 4.3 depicts the autocorrelation as a function of the time shift s .

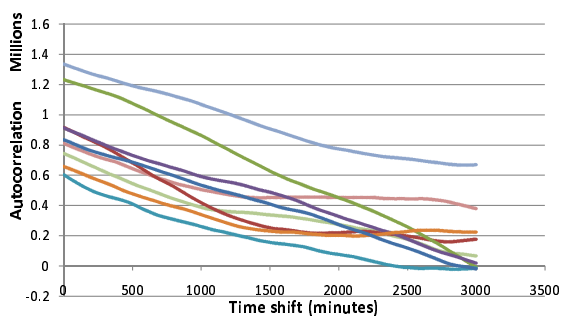
The figure shows that for real traces, there are sharp rises in autocorrelation individually and on average when the time shift is one day. The bus traces also show repeatedly oscillating autocorrelation values throughout a day because each bus runs on a periodic schedule. Such oscillations are much less obvious for the cabs as they move more randomly in the city.



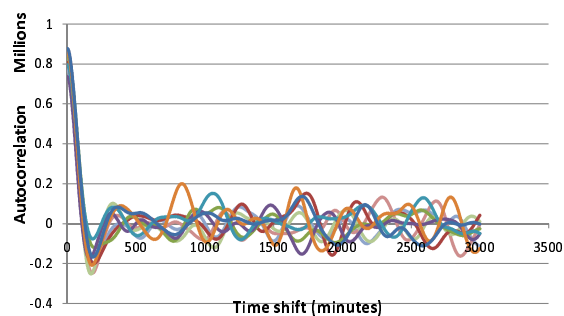
(a) Samples for cab traces



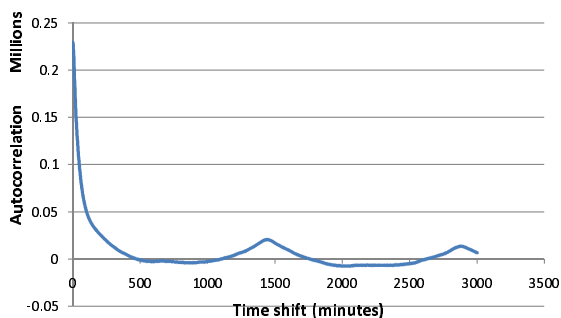
(b) Samples for bus traces



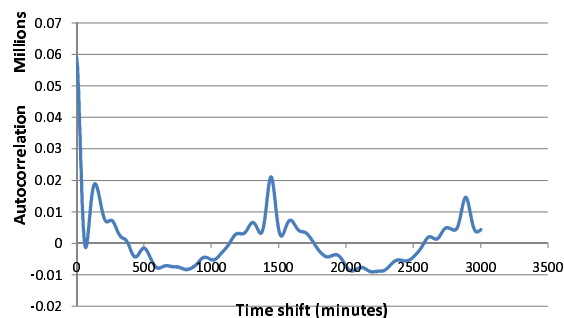
(c) Samples for random walk



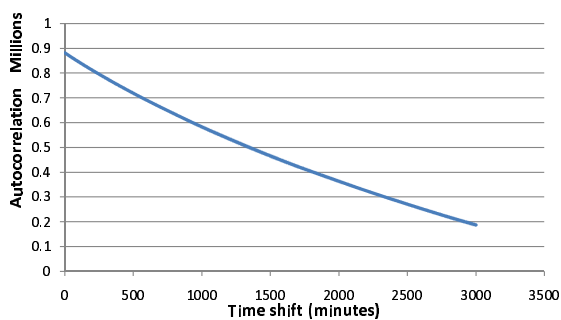
(d) Samples for random waypoint



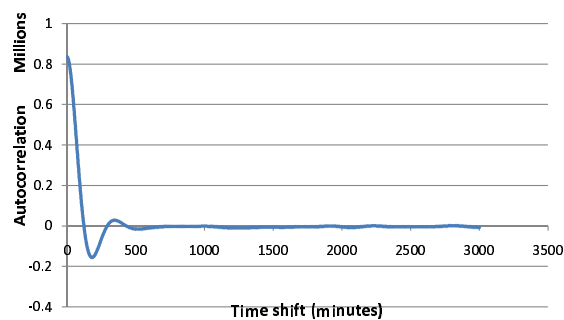
(e) Average for cab traces



(f) Average for bus traces



(g) Average for random walk



(h) Average for random waypoint

Figure 4.3. Autocorrelation of each trace for different sets of traces as a function of the time shift s .

Complexity of movement

In this section, we demonstrate the complexity of nodal movements as quantified through the order- n model complexity given by:

$$H_n\left(\{L_i\}_{i=1,2,\dots,N}\right) = \sum_{\Theta^{n-1}} p(l_1, \dots, l_{n-1}) \times \sum_{\Theta} p(l_n | l_1, \dots, l_{n-1}) \log p(l_n | l_1, \dots, l_{n-1}) \quad . \quad (4.10)$$

In the above, the functions $p(\dots) : \Theta^{n-1} \rightarrow \mathbf{R}_+$ and $p(\cdot | \dots) : \Theta \times \Theta^{n-1} \rightarrow \mathbf{R}_+$ are the joint probability and conditional probability densities of the locations in the collection of traces $\{L_i\}_{i=1,2,\dots,N}$.

The above function, defined for general stochastic processes, is well-known in the information theory community (see for example [57, Chapter 3]). The value of H_n represents the uncertainty of the order- n model. The smaller the value, the less uncertainty there is in the model. Notice that H_0 is essentially the entropy of the stationary distribution.

The behavior of (4.10) as a function of n is depicted in Fig. 4.4. The result conforms to the theoretical result that for any stationary process X , $H_n(X)$ is a decreasing function of n and the limit $\lim_{n \rightarrow \infty} H_n(X)$ thus exists. The limiting value is called the *entropy rate* of the process X .

Notice that because of the relatively slow movement of the mobile nodes, the synthetic traces do not have enough time to reach steady state if we limit the synthetic traces to the same length as the real traces in quantifying their characteristics. As a result, although one may expect random walk traces to have a constant entropy as the order increases, we observe otherwise in the figure, and the entropy of the synthetic traces also drops more significantly than the real traces as the order increases.

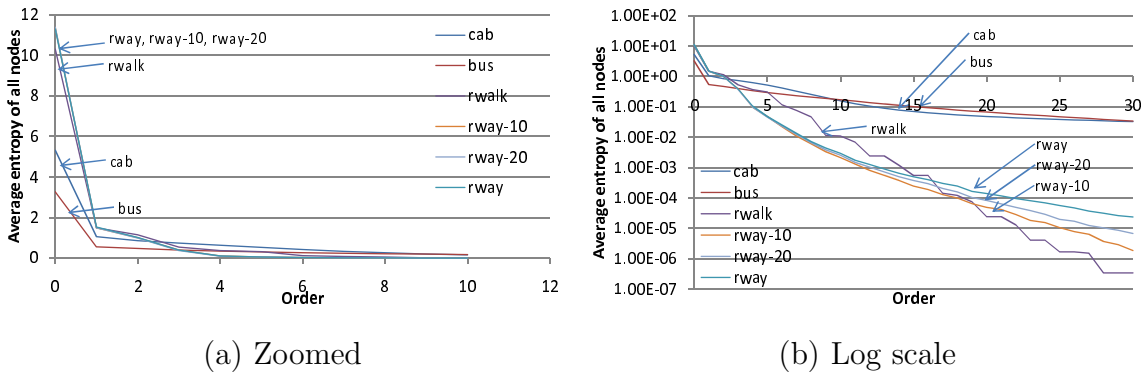


Figure 4.4. Order- n complexity of different sets of traces as a function of order.

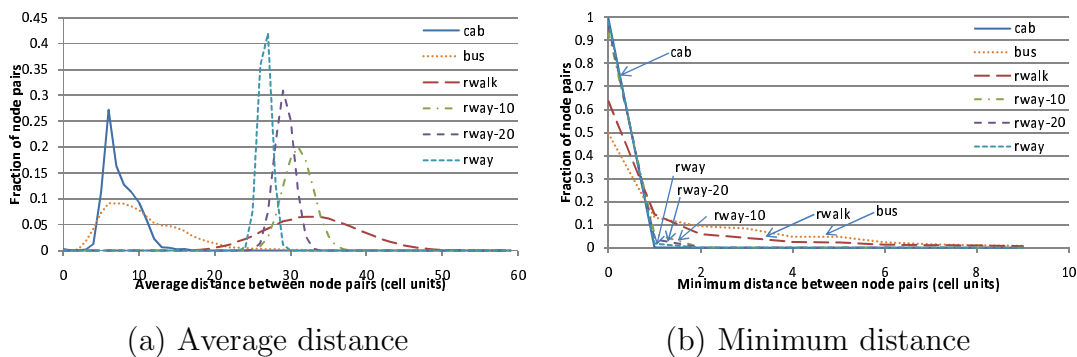


Figure 4.5. Average and minimum distances between each node pair.

Distance between traces

Fig. 4.5(a) depicts the distribution of average distance between trace pairs, which is defined as

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N |L_i(k) - L_j(k)|,$$

for trace pair i and j , where $i \neq j$, and Fig. 4.5(b) depicts the distribution of minimum distance between trace pairs, which is defined as

$$\min_k (|L_i(s_k) - L_j(s_k)|).$$

Implications of the trace characteristics

Many of the observed different characteristics of the mobility traces can be summarized and explained using the lack of *preferred locations* and the random initial positions of the synthetic traces. Specifically, real traces show natural preferences for certain places visited by the mobile nodes, such as the busy downtown area for the cabs and the assigned routes for the buses, whereas the synthetic traces do not.

Since synthetic traces do not have a set of preferred locations to visit, and they are placed randomly in the network initially, they exhibit much sparser spatial distributions in the network than the real traces. Moreover, the shorter the maximum trip length in the synthetic traces, the sparser they are in the network. These observations are shown through the higher order-0 entropy of the synthetic traces than the real traces (Fig. 4.4), the larger autocorrelation values of the synthetic traces (Fig. 4.3), the longer average distances between synthetic node pairs (Fig. 4.5(a)) (with random walk having the largest average distance), and the fact that not all the random walk nodes have met each other during the simulation (Fig. 4.5(b)). On the other hand, the real traces have their preferred visiting places, resulting in a smaller H_0 than the synthetic traces, and the entropy drops much more slowly when the order increases. This is also reflected in the smaller average distances between cabs or between buses, although the distance between buses exhibits a broader range because some of the bus routes are closer together while some are farther apart. The result is that only 50% of the buses have met each other as shown in Fig. 4.5(b), while almost 100% of the cabs have met each other.

When nodes are more sparsely distributed in the network area, more efficient victim identification results when the adversary collects the side information passively. Hence, we would expect victims in the synthetic traces to be more easily identified than in the real traces. However, sparsity of nodes can both be beneficial and detrimental to the performance of an adversary who observes the participants directly. It is because when the mobile nodes are sparsely distributed, it could take much longer

time for the adversary to meet them, thus harming the attack efficiency. On the other hand, once the adversary meets a mobile node, it could identify the trace of the node almost instantaneously as no other mobile nodes (and hence, traces) are around at the same time, thus helping the attack performance. We will verify these expectations experimentally in the following section.

4.4.2 Passive Adversary

In this section, we study the attack scenario where the adversary tries to identify the trace of one participant (the victim) by gathering side information passively. In each simulation, the victim is randomly picked from all the participants. Pairs of $\langle \text{time, location} \rangle$ of the victim are then randomly sampled from the trace and noise is introduced in the spatial domain. The noisy data are revealed to the adversary as side information, which the adversary utilizes to identify the complete movement history of the victim from the published traces. Results reported are for simulation experiments each repeated 100,000 times.

We quantify the performance of the strategies with the following metrics,

1. *Fraction of correct conclusions.* A conclusion is correct if the victim is uniquely identified according to the criterion of highest similarity metric.
2. *Fraction of incorrect conclusions.* A conclusion is incorrect when the victim is not among the set of candidates having the highest similarity metric.

Problem A1 - side information references time instants that coincide with sampled times in the trace

We present the results based on the perception of the adversary on the noise.

(1) Correct assumption of noise distribution

We first consider the case when the revealed location of the victim is perturbed with zero-mean Gaussian noise with standard deviation σ , which matches the as-

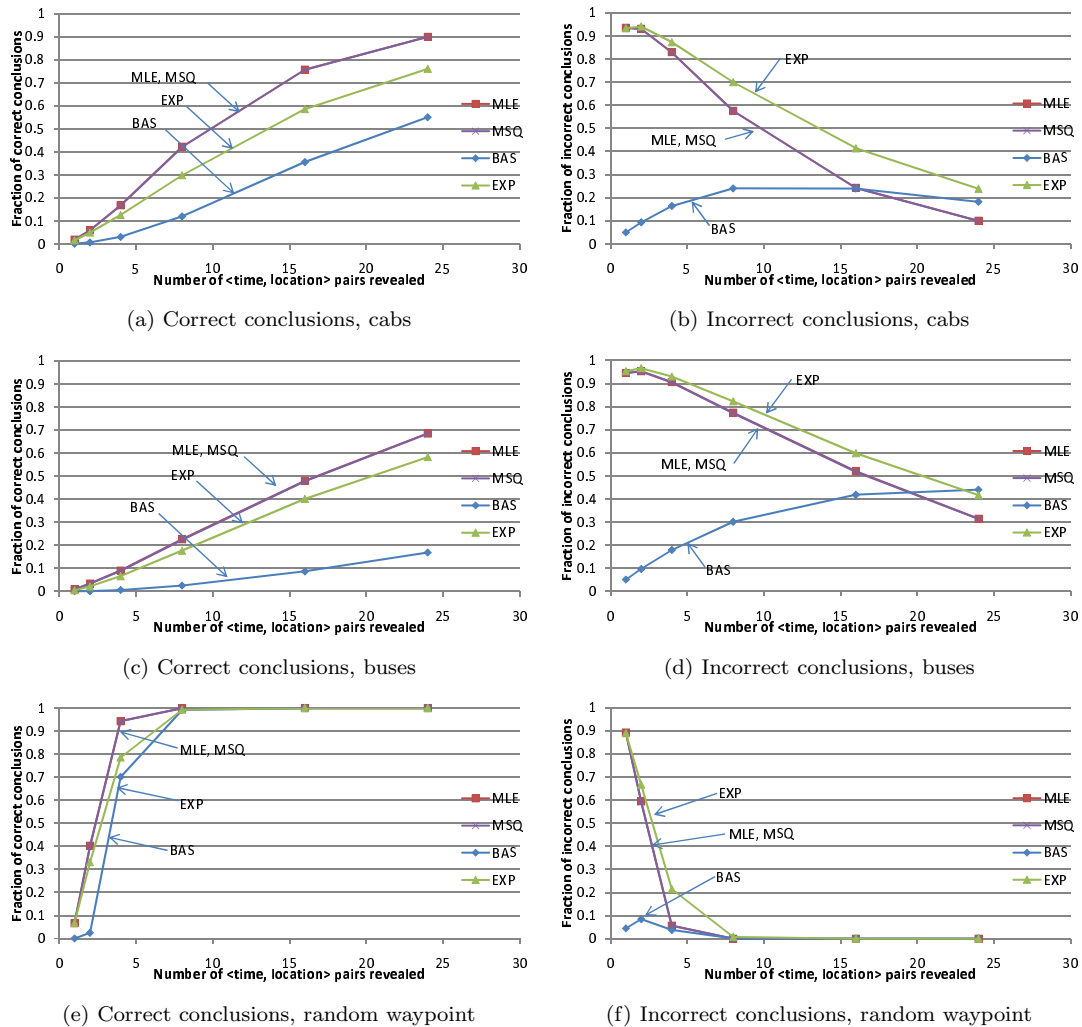


Figure 4.6. (A1) Performance of various metrics as a function of the number of $\langle \text{location}, \text{time} \rangle$ pairs revealed. (a), (b) San Francisco cab traces, (c), (d) ShangHai Grid bus traces, and (e), (f) random waypoint traces. Zero-mean Gaussian noise with $\sigma = 5$.

assumption made by the adversary in **MLE**. Fig. 4.6 shows the performance of the attack strategies using the cab, bus, and random waypoint traces. Results of random walk traces are not shown because they give similar trends as random waypoint.

When we compare the two attack strategies that assume knowledge of the noise, namely **MLE** and **BAS**, **MLE** is more aggressive as it excludes a trace from further consideration as soon as it determines that the trace cannot be perturbed to the revealed locations of the victim given the type and magnitude of the noise assumed.

Hence, when the adversary’s assumption is correct, this approach can give very good results in the fraction of correct conclusions, although it can also give a large fraction of incorrect conclusions initially, when the adversary has only a few pairs of the side information. In comparison, **BAS** generally returns lower fractions of both correct and incorrect conclusions as it gives equal weights to traces that agree with the side information within the error bounds. This results in more undecided conclusions.

We now look at the other two approaches that do not use knowledge of the noise, namely **MSQ** and **EXP**. We can see that although **MSQ** does not require the knowledge, its performance is similar to the best-case performance of **MLE** in terms of the fraction of correct conclusions. Meanwhile, **EXP** performs the worst as it puts too much weight on traces that give little deviations from some of the pieces of side information.

(2) Wrong noise assumptions

We now consider the case when the assumption of noise distribution made by the adversary in **MLE** is incorrect. Figs. 4.7(a) and (b) show the performance of the strategy when the actual and assumed noise is Gaussian and Uniform, respectively. Figs. 4.7(c) and (d) show the results when the actual and assumed noise is Uniform and Gaussian, respectively. Figs. 4.7(e) and (f) show the results when the noise distribution is Uniform, and the adversary assumes the same.

Notice that among the approaches that assume about the noise, **MLE** is affected the most by the wrong assumptions. In particular, the performance of **MLE** varies depending on the types of actual and assumed noise. When the adversary assumes the noise to be Uniform but it is Gaussian, the performance much worsens since the victim’s trace can be mistakenly and permanently removed from consideration due to occasional Gaussian noise that exceeds the range of the assumed Uniform noise. On the other hand, when Gaussian noise is assumed but it is actually Uniform, **MLE** surprisingly gives a greater fraction of correct conclusions than when the correct noise distribution is assumed, albeit at the price of getting a greater fraction of incorrect

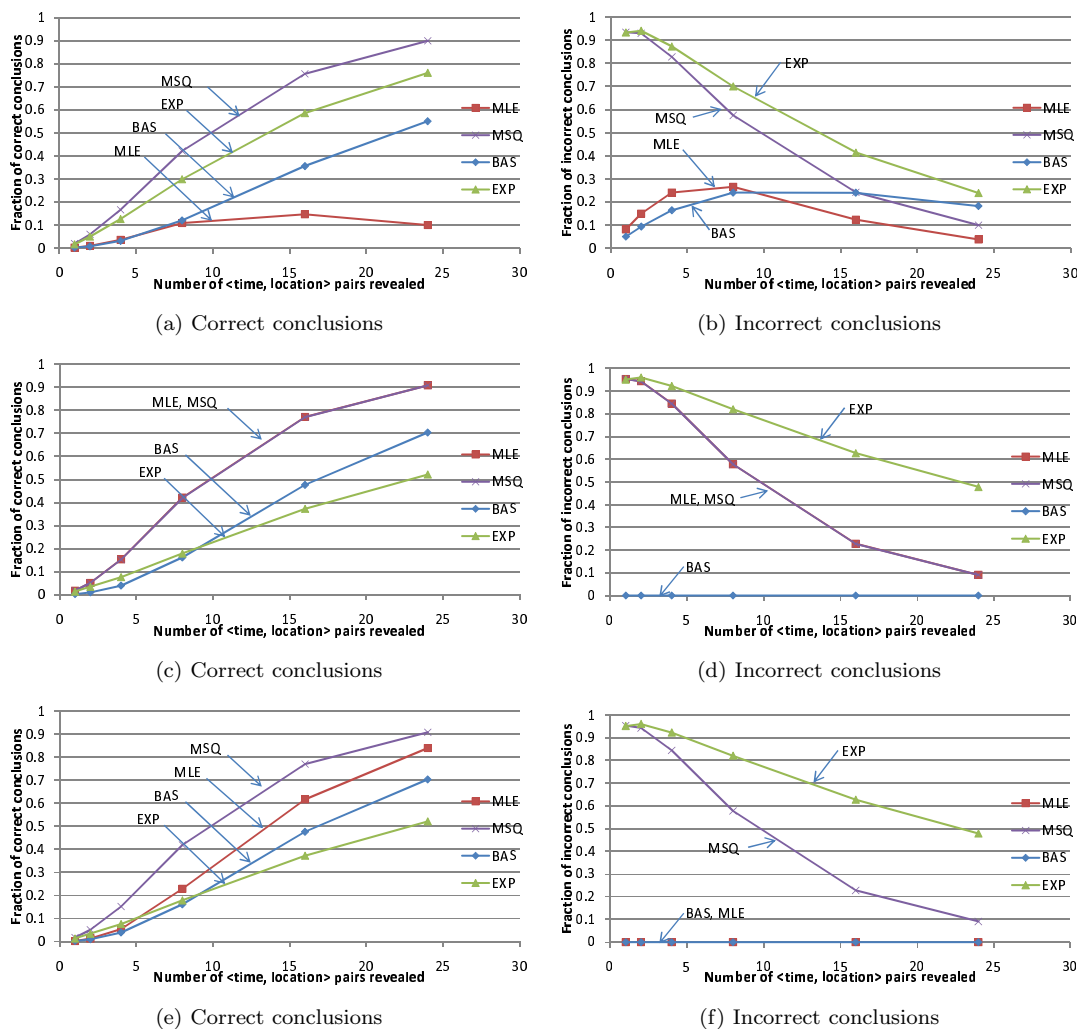


Figure 4.7. (A1) Performance of various metrics as a function of number of <location, time> pairs revealed. (a), (b) Uniform noise assumed, Gaussian actual; (c), (d) Gaussian noise assumed, Uniform actual; (e), (f) Uniform noise both assumed and actual. San Francisco cab traces. Noise with $\sigma = 5$.

conclusions also. In contrast to **MLE**, the performance of **BAS** is less sensitive to the type of noise.

Problem A2 - side information references time instants that does not coincide with sampled times in the trace

Fig. 4.8 depicts the performance of the attack approaches for different sampling time intervals. Zero-mean Gaussian noise with $\sigma = 5$ is introduced into the spatial domain of the side information except for the line labeled “no noise.” The figure shows that the sparser the samples in the traces, the less effective the attacks are in general. This is expected since when samples are sparser, inference of nodal movements between the sampling points becomes less reliable. Fig. 4.9 depicts the results for the bus traces and the synthetic random waypoint traces. The figure shows that without noise in the side information, even with a sampling temporal granularity of an hour and spatial granularity of 0.01° , the adversary is able to identify the victim’s trace by fewer than 25 pairs of side information with high probability. When noise is introduced, however, the results depend heavily on the traces. For instance, the effect of noisy side information on the attack strategies is insignificant for the synthetic traces, but it is more noticeable for the bus traces.

When we compare the performance of the attack approaches in this case with the special case in the previous subsection, in which no inference using a general movement model is necessary, the performance here does not degrade significantly for **MLE**₂ and **MSQ**₂. Interestingly, **BAS**₂ gives a large fraction of correct and incorrect conclusions initially when movement has to be inferred, while **EXP**₂ performs about the same.

Summary on passive adversary strategies

The results show that approaches relying on the assumption of noise could have very poor performance when the assumption is wrong, as illustrated by the **MLE** results. On the other hand, an approach not having knowledge of the noise may still perform well. In particular, **MSQ** performs equally well as **MLE** even when the latter has the correct noise assumption. Since **MSQ** also performs better than

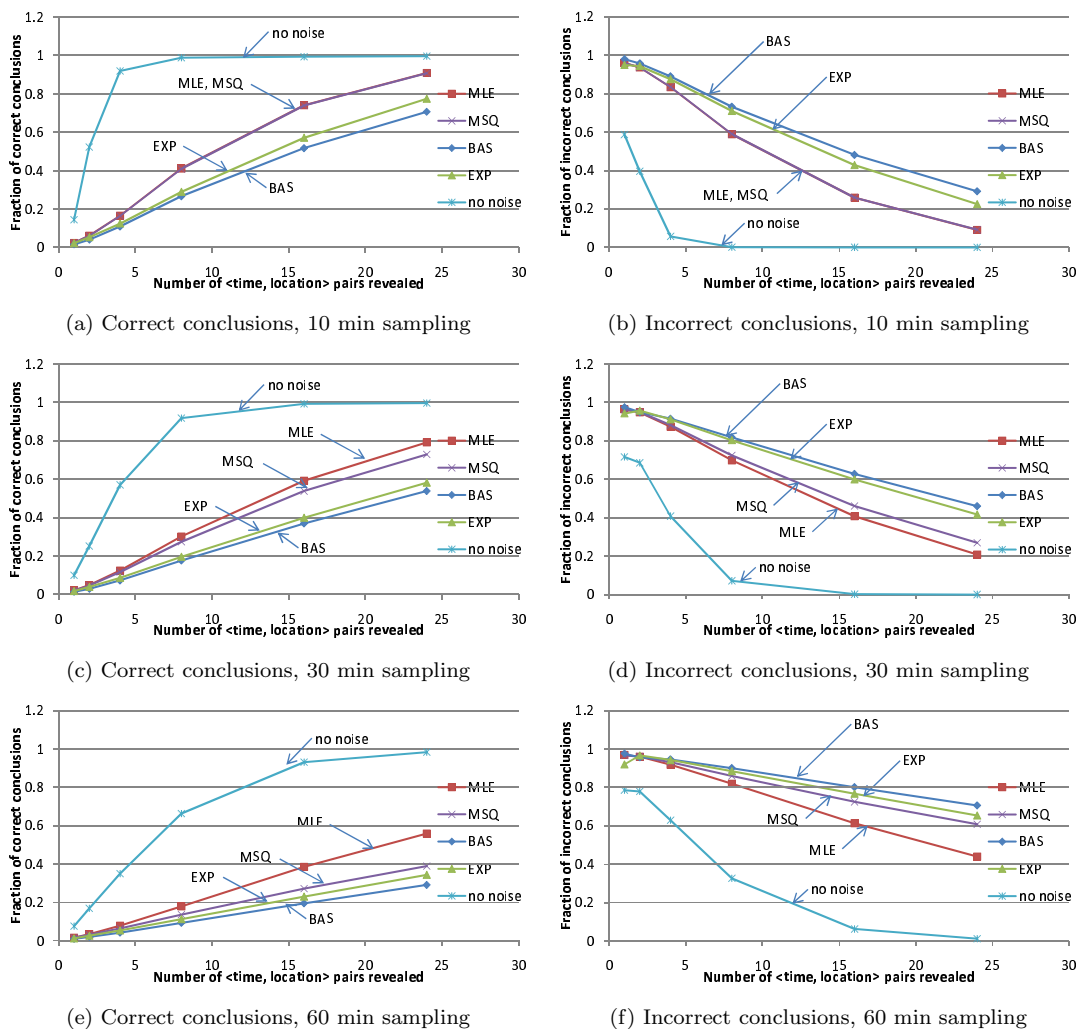


Figure 4.8. (A2) Performance of various metrics for attacks requiring different degrees of movement inference for each trace as a function of number of $\langle \text{time, location} \rangle$ pairs revealed. San Francisco cab traces, zero-mean Gaussian noise with $\sigma = 5$. (a), (b) $S =$ ten minutes; $T =$ five minutes; (c), (d) $S =$ thirty minutes; $T =$ fifteen minutes; (e), (f) $S =$ one hour; $T =$ half an hour. (S is the trace sampling time and T is the interval for computing the transition matrix.)

the heuristic approaches of **BAS** and **EXP**, it appears to be the preferred adversary strategy overall.

The results also verify our claim in Section 4.4.1 that victim identification is much easier for the synthetic than real traces, due to higher nodal sparsity in the former.

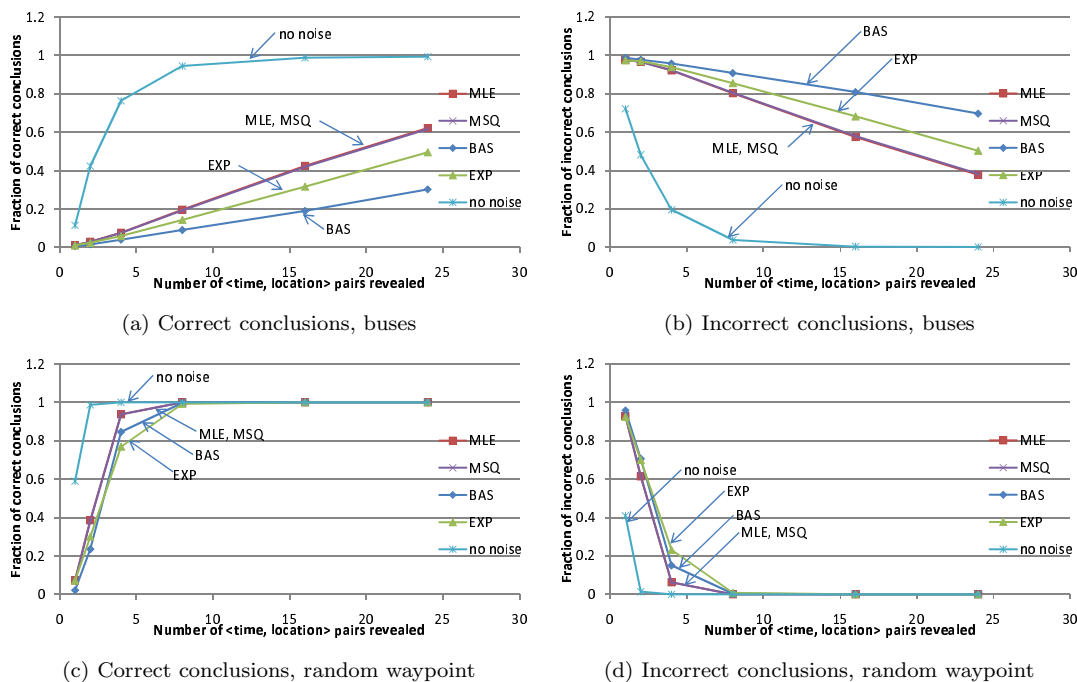


Figure 4.9. (A2) Performance of various metrics for attacks requiring different degrees of movement inference for each trace as a function of number of $\langle \text{time}, \text{location} \rangle$ pairs revealed. Traces are sampled every half an hour and the transition matrix is generated using sampling information every fifteen minutes.

4.4.3 Results for Active Adversary

In this section, we examine the performance of the active adversary which gains side information by direct meetings with the participants. Recall that this adversary can identify a victim by elimination, and the process is most efficient if the adversary meets the participants as quickly as possible. We assume that the adversary operates to achieve this goal. We further assume that the adversary's side information is gained only at times coinciding with sampled times of the traces. As discussed in Section 4.4.1, we expect the active adversary needs a longer time to identify all the synthetic traces than the cab traces, because the former have sparser node distributions. Further, we expect random walk requires the longest time among the synthetic

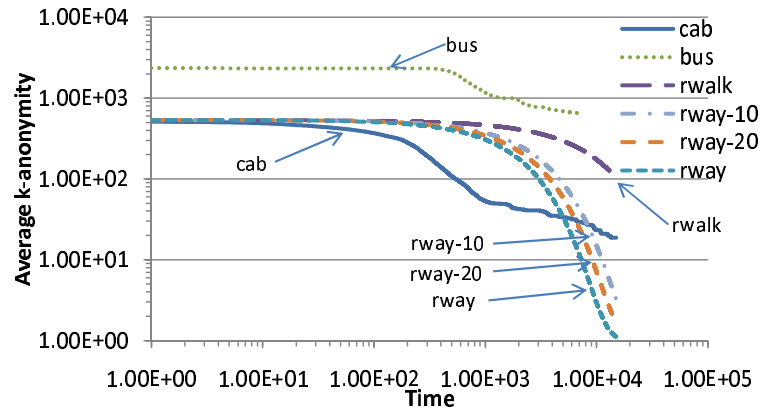


Figure 4.10. (B1) Average k -anonymity as a function of attack time when the adversary is one of the mobile nodes.

traces, and the bus trace requires the longer time between the real traces. To quantify the performance, we consider the following five metrics when applicable,

- *k -anonymity as a function of time.* This metric represents the number of traces the victim's one cannot be distinguished from using the side information available to the adversary.
- *The number of encounters to identify a victim.* This is the total meeting time (in terms of time steps) to identify a victim's trace.
- *The number of visits to identify a victim.* This is the total number of non-consecutive meetings with a victim to identify the victim's trace.
- *The length of time from the beginning of simulation to identify a victim.* This is the duration to identify a victim from the beginning.
- *The length of time from the first encounter to identify a victim.* This is the duration to identify a victim from the first meeting with the victim.

Table 4.2

Statistics of an active adversary who is one of the participants to identify a victim.

Traceset	Average number of encounters	Average number of visits	Average time (mins) from beginning	Average time (mins) from first encounter
cab	18.486	2.064	967.589	142.430
shbus	12.443	1.736	1538.863	301.192
rwalk	1.087	1.020	6228.523	11.074
rway-10	1.077	1.018	4458.226	44.106
rway-20	1.092	1.023	3965.484	57.673
rway	1.119	1.031	3382.869	71.222

Problem B1 - adversary as one of the participants

Fig. 4.10 depicts the average k -anonymity of the victims as observed by the adversary as a function of the attack time for different sets of the traces, when the adversary is one of the mobile nodes. The figure shows that the most reduction in k -anonymity for each participant results from observations made in the first day in the real traces. The figure also shows that as time increases, the k -anonymity always drops to close to one on average, except for random walk traces and the bus traces as expected.

Table 4.2 lists the averages of the other four metrics, which shows that, as our expectation, synthetic traces takes fewer number of encounters and visits to identify a victim trace. Among random waypoint traces, the set without bound takes slightly larger number of encounters to identify a victim trace because the mobile nodes are more likely to be present at the center of the network area, and meet with each other there with the presence of other nodes. Hence, it is harder for an adversary to identify the victims, and more encounters are needed (albeit slightly more). On the other hand, random walk traces also need slightly more encounters and visits to identify a victim trace than other synthetic traces because of the synchronized movement of the nodes, which means that if two mobile nodes visit the same cell at

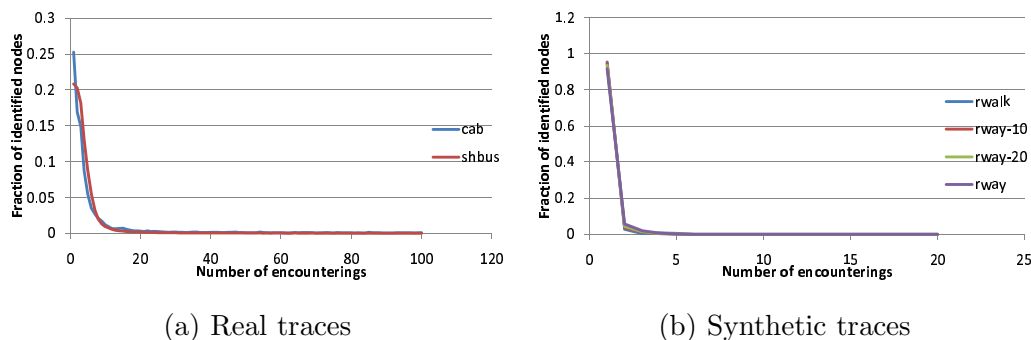


Figure 4.11. Distribution of total number of meeting time to identify a victim when the adversary is one of participants.

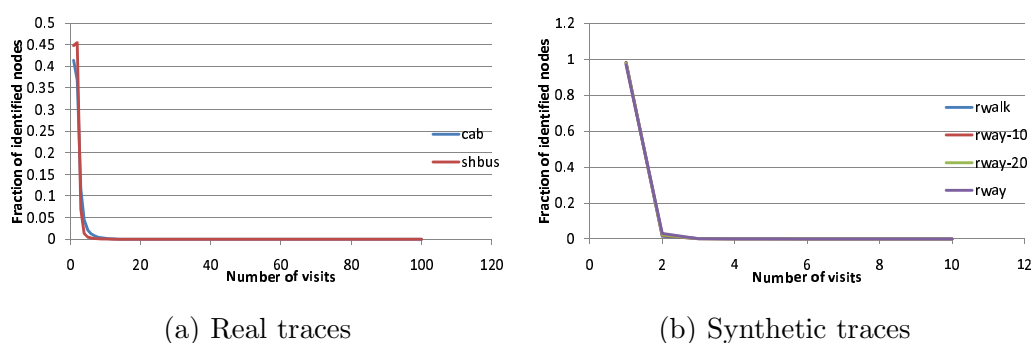


Figure 4.12. Distribution of number of non-consecutive meetings to identify a victim when the adversary is one of participants.

the same time together with the adversary, they are likely to leave the cell at the same time, and the adversary needs at least one more encountering to differentiate the two.

The table also shows that cab traces need more encounters and visits than bus traces to identify, but the total time needed is shorter. It is because when the adversary is one of the cabs, they are more likely to follow the same path for a longer time in the busy district. At the same time, when cab traces cannot be identified, they can be encountered more quickly in the busy district. On the other hand, buses are more likely to follow different routes, and hence, they are less likely to encounter each other consecutively for long time. Moreover, since they run different routes, they are met much less frequently than cabs, and hence, it takes a longer time to identify them.

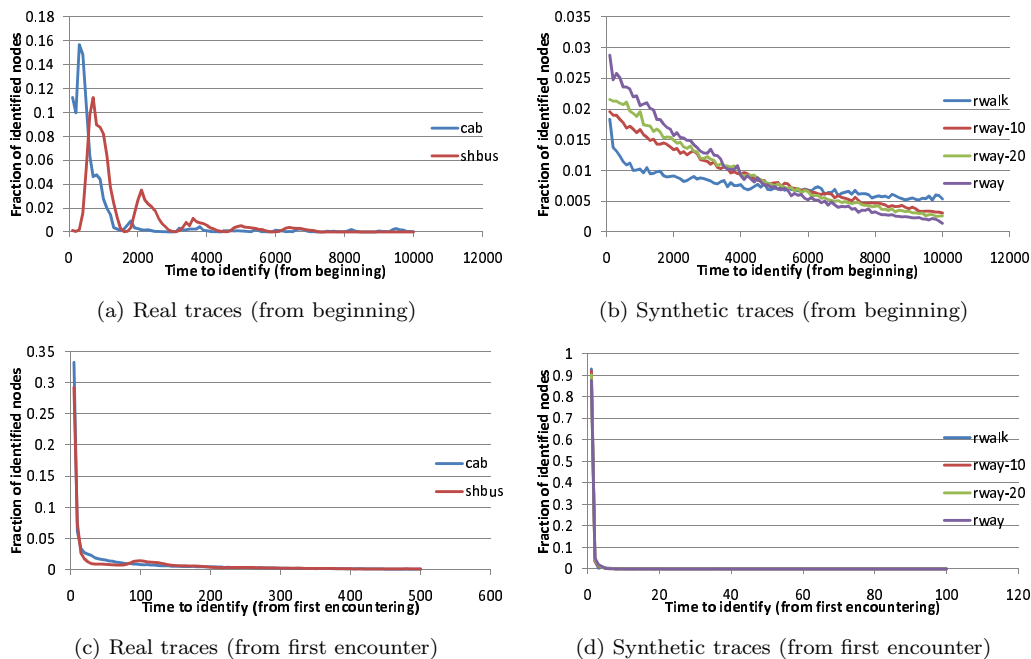


Figure 4.13. Distribution of time ((a), (b) from beginning; (c), (d) from first encounter) to identify a victim when the adversary is one of participants.

Fig. 4.11 depicts the distribution of total number of meeting time to identify a victim, while Fig. 4.12 depicts the distribution of number of non-consecutive meetings to identify the victim. They show that if the adversary is one of the participants in real traces, it takes a long time for him to identify a victim. It is because in a short time interval, nodes that are close together are more likely to stay together because of physical constraints or preferences. This is much less likely for synthetic traces.

Fig. 4.13 depicts the distribution of time from the beginning of the simulation and from the first encounter to identify a victim. The figure shows that for real traces, there are periodic pattern in the number of victims identified as a function of time from the beginning. The sharp peaks of the buses indicate the early morning peak hours when buses start serving. The pattern for cabs is less obvious as some of them work overnight. Among all synthetic traces, shorter time from beginning is needed to identify a trace of mobile nodes of longer trip length. It is because when nodes can

Table 4.3
Statistics of a static adversary to identify a victim.

Traceset	Average number of encounters	Average number of visits	Average time (mins) from beginning	Average time (mins) from first encounter
cab	1.569	1.040	4427.944	20.454
shbus	3.929	1.064	1556.462	19.611
rwalk	1.112	1.018	6649.473	12.003
rway-10	1.065	1.004	5746.883	19.100
rway-20	1.074	1.005	5406.299	27.807
rway	1.084	1.006	5152.405	31.429

travel a longer distance, they are more likely to visit more places in a shorter time and possibly meet with the adversary.

The figure also shows that for synthetic traces, most of the victim traces can be identified in a short time. In particular, most of the victim traces can be identified at the first meeting with the adversary because there are no other mobile nodes at the same location at the same time when the victim meets with the adversary.

Problem B2 - adversary stays at one of the cells

Fig. 4.14 depicts the k -anonymity of the victims as observed by the adversary as a function of attack time, when the adversary stays at one of the cells. Each line in the figure represents the results for a particular staying cell, and the line label shows the relative coordinates of that cell in the network area. We plot the results of the six most popular cells in each figure, and the popularity of a cell is ranked according to the total number of visits made by the mobile nodes over the entire trace.

The figures indicate that for the real traces, staying at a cell for a day is sufficient to reduce the k -anonymity for each participant significantly. The improvement by staying longer at each cell is minimal. The k -anonymity of the random walk and bus traces drops more slowly than the other traces as expected.

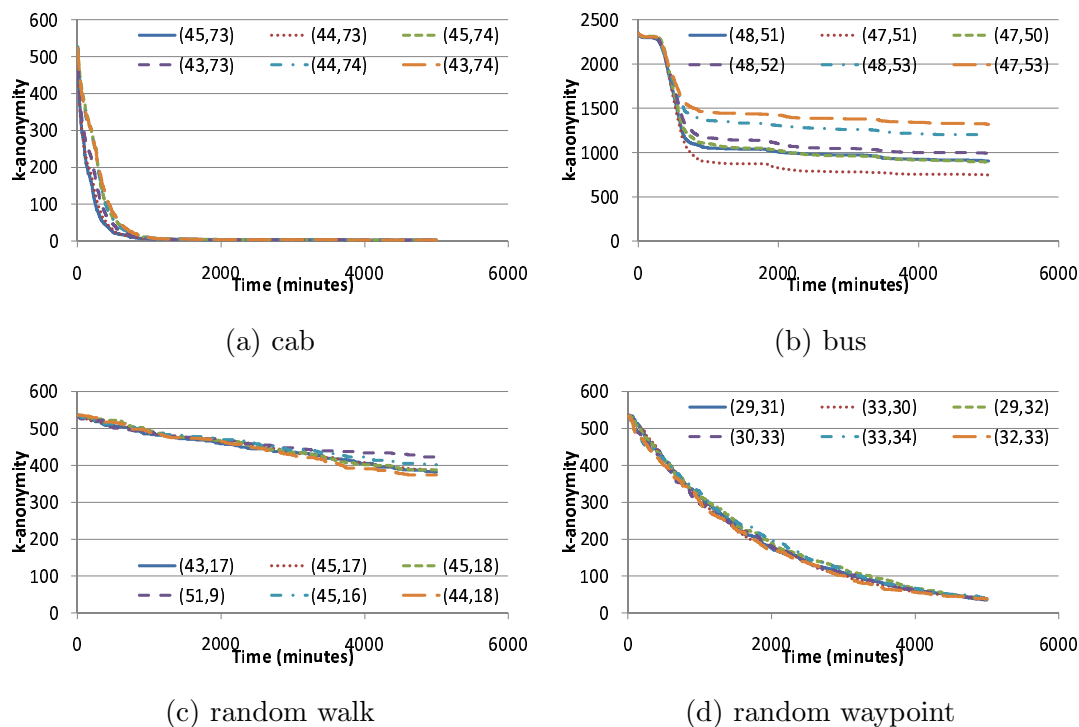


Figure 4.14. (B2) k -anonymity of the victim as observed by a static adversary as a function of attack time.

Table 4.3 lists the averages of the other four metrics, which shows that the performance for synthetic traces when the adversary is one of the participants is very similar to the previous case. It is because for synthetic traces, being one of the mobile nodes is very similar to being one of the cells.

At the same time, the table shows that it takes longer time to identify a victim from synthetic traces than real traces from the beginning of the trace collection period. It is because synthetic mobile nodes are more sparsely distributed in the network area, and the time for a mobile node to move across the area is longer when they are only able to travel a shorter distance. Notice that among the real traces, bus traces take shorter time to identify. It is because buses usually follow a specific schedule throughout a week. Hence, it is more likely for a static adversary to identify the trace of a bus on the first day, or the adversary cannot identify the victim's trace at all since the bus does not visit the cell during the week.

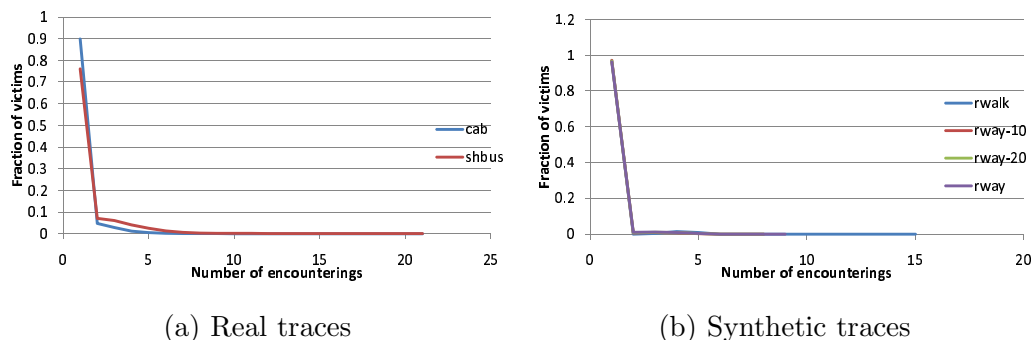


Figure 4.15. Distribution of number of encounters to identify a victim when the adversary is static.

The table also shows that when we consider the time needed to identify a victim trace from the first encountering among synthetic traces, the one with shorter trip length takes shorter time. It is because if the mobile node cannot be identified instantaneously, it takes a shorter time for them to reach the same cell again if they can only travel for a shorter length, and then they are very likely to be identified.

Fig. 4.15 depicts the distribution of number of encounters to identify a victim, which shows that fractionally, most of the synthetic traces can be identified by encountering the mobile node once. The performance among synthetic traces is very similar. While among real traces, interestingly cab traces are more easily identified than the bus traces in terms of the number of encounters. Intuitively we would expect the otherwise because cabs are more likely to move within the downtown area, and hence, they should be less easily identified since there are more traces show similar paths. On the other hand, as cabs are less likely to follow each other for a long time, when they cannot be identified at the first encountering by the adversary, they are more likely to be differentiated at the next encountering. Meanwhile, it takes a longer time for buses to move across a cell, resulting in a greater number of encounters before the trace of a passing node is identified.

Fig. 4.15 also shows that when we consider the number of victims identified among the synthetic traces, the longer the average trip length, the greater the number of victims identified by a static adversary with a single observation, except the one without

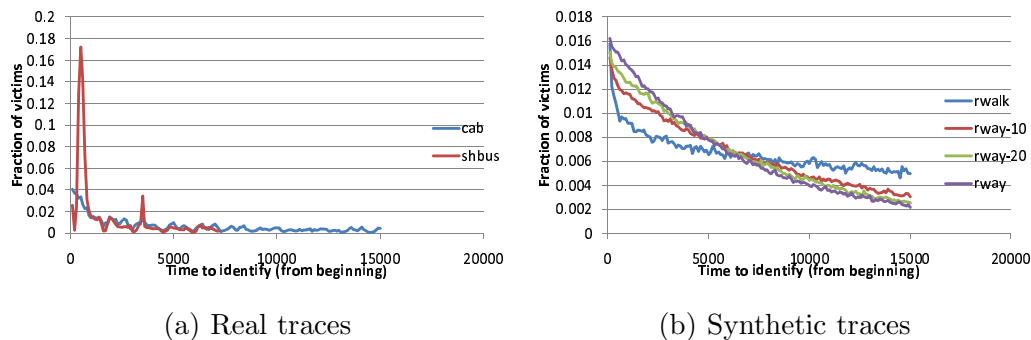


Figure 4.16. Distribution of time (from beginning) to identify a victim when the adversary is static.

bound in trip length. It is because when the trip length increases, a mobile node is more likely to visit more places, resulting in greater number of victims identified (recall that the figure considers all victims identified at all possible locations.) When there is no bound on the travel trip length, however, cells at the center are more likely to be visited by more than one mobile nodes at the same time, and hence, victims are less easily identified there, resulting in a slight drop in the number of victims identified.

Fig. 4.16 depicts the distribution of time from the beginning of the traces to identify a victim, which shows that for real traces, there are periodic pattern in the number of victims identified as a function of time. The sharp peaks of the buses indicate the early morning peak hours when buses start serving. The pattern for cabs is less obvious as some of them work overnight. Among all synthetic traces, shorter time is needed to identify a trace of mobile nodes of longer trip length. It is because when nodes can travel a longer distance, they are more likely to visit more places in a shorter time, and being identified at these locations earlier.

Problem B3 - adversary moves actively in the area

Fig. 4.17 depicts the k -anonymity of the victim as observed by the adversary as a function of attack time, when the adversary moves actively inside the network area.

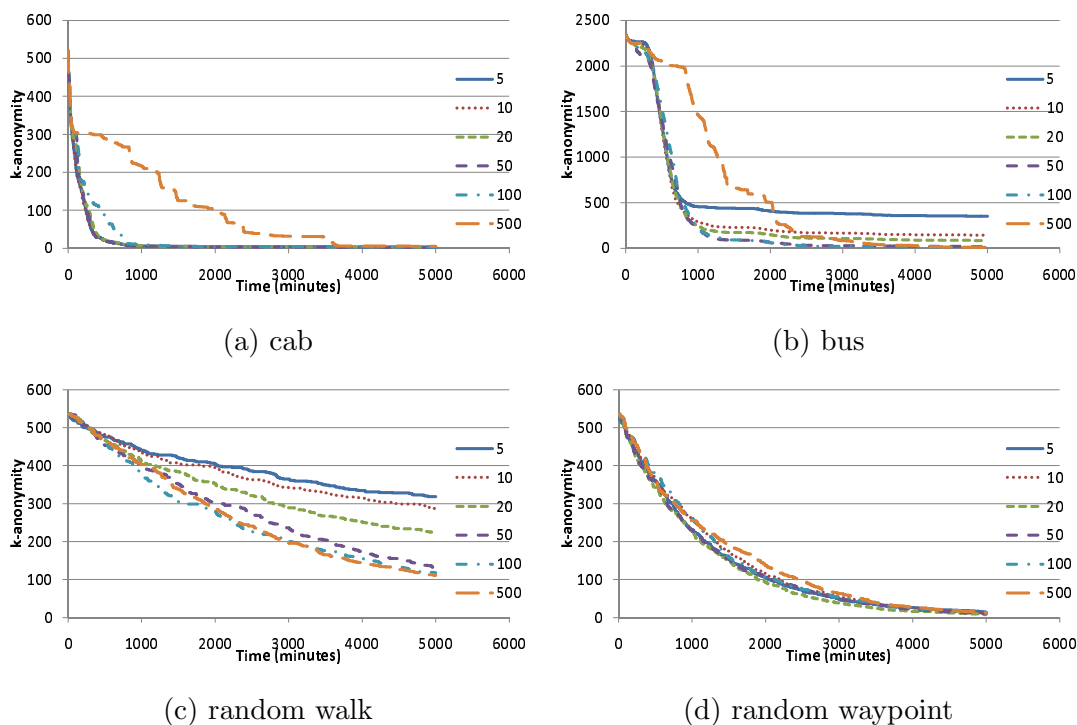


Figure 4.17. (B3) k -anonymity of the victim as observed by the adversary as a function of attack time, when the adversary is mobile within a pre-determined path.

The label of each line in the figure indicates the number of popular cells visited by the adversary. Notice that as the adversary travels between the popular cells, it may visit other cells during the journeys.

The figures show that travels made by the adversary generally improve the attack efficiency in identifying the traces. For instance, for the bus traces, traveling helps the adversary reduce the size of the candidate set for each participant from more than 2000 to only a few in about one day, while staying at a cell can only reduce the size by half. It is because by traveling, the adversary is able to meet more participants, especially when their spatial distribution is sparser, such as the random walk and bus traces. However, traveling to too many places may hurt the performance because the adversary may spend too much time traveling over unpopular places.

Summary on active adversary strategies

The results show that for the real traces, the ability of the active adversary to travel helps it identify many of the victim traces within one day. For synthetic traces, however, the attack efficiency is lower because their spatial distribution is sparser, verifying our observation in Section 4.4.1, though it is still better than staying at a cell or being one of the participants in general. When the adversary prefers to stay at a cell, the attack efficiency depends on the type of traces and the location of the adversary. In general, staying at a more popular location helps by allowing the adversary to identify more victims more quickly.

4.5 Summary

In this chapter, we discussed the privacy vulnerability of publishing mobile traces to assist the analysis and evaluation of mobile sensor networks, since the mobile entities could also be observed directly or indirectly by an adversary, who tries to identify the complete path history of one or more of the participants. We presented comprehensive strategies for an adversary to utilize side information about node movements to achieve different privacy attacks. The simulation results under comprehensive system parameters, such as the nodal mobility, adversary strategy, noise in the trace or the side information, and different degrees of movement inference needed for the attack, show that the adversary is able to identify victims with high probability even when the available side information is limited and noisy.

We also explained the differences between traces with different characteristics, such as the different sparsity of real and synthetic traces, from the perspective of the privacy problem. A privacy protection measure targeting the synthetic traces may not be effective on real traces because of these differences.

5 SUMMARY

In this dissertation we have studied two interesting and important problems in mobile sensor networks.

We have studied mobility algorithms for sensors to cover a surveillance area. We considered two types of mobility: stateful and stateless. For the stateful algorithms, we studied how to add features to a basic algorithm to achieve a good balance between different (possibly antagonistic) goals of the user. We showed the benefits of pausing in the weighted random waypoint algorithm for achieving (i) good matching between the coverage and threat profiles of the network, and (ii) high-confidence event monitoring when the temporal dimension of sensing is considered. Further to single sensors, we studied various coordination approaches for multiple sensors. We found that if the sensor density is low, or if matching and fairness are the only performance objectives, then coordination among the sensors is largely not necessary. However, we found also that if the sensor density is high, then different coordination approaches can result in significantly different performance, complexity, and robustness to changes. In particular, to have good matching, exposure, and effective coverage simultaneously, the sensors should partition their responsibilities for cover different parts of the network, such that each sensor will only need to monitor a smaller sub-region.

For the stateless mobility algorithm, we analyzed the use of a steepest descent algorithm to compute the optimal transition probabilities for maximizing a given utility function. We further proposed the use of an adaptive time step to improve the search efficiency as an implementation issue. Importantly, our results demonstrate that numerous local optima may exist in the solution space, and we showed how the use of controlled noise can help us get out of these local optima with high probability.

We have analyzed the quality of monitoring of a general paradigm of periodic coverage by mobile sensors. We showed the basic importance of the temporal dimension

of sensing in determining the performance of different coverage strategies. Our results are also applicable for duty-cycled static sensors that are widely deployed for energy efficiency.

We then studied the privacy issue in publishing anonymous mobility traces. We provided a comprehensive study on analyzing available attack strategies to an adversary who may collect or infer location information of one or more participants in the network. We formally proved that certain strategies can use all available information effectively in the attack. Our analysis is verified and complemented by simulation results under comprehensive system parameters, including nodal mobility, adversary strategy, noise in the trace or the side information, and degree of movement inference needed. In particular, we studied different characteristics of synthetic and real-world mobilities, and discussed their implications on the privacy attack. In general, our results showed that the adversary is able to identify the complete history of the victims with high probability even when the available side information is limited, indicating the privacy concerns of publishing these trace sets.

5.1 Future Work

In this dissertation we have presented algorithms for controlling the movement of mobile sensors for given user- and application-specific goals. Further interesting research includes the following.

- **Better coordination between multiple sensors.** We have studied a number of coordination approaches that vary in performance complexity. Algorithms that are provably optimal by certain metrics are still missing. Also, the way responsibilities are divided among the sensors in the MD and MDO approaches is specific to the goals of matching and fairness. It is interesting to study how responsibilities can be divided effectively when other performance considerations are introduced into the cost function.

- **Stateless mobility algorithm for multiple sensors.** The stateless mobility algorithm is based on the optimization of a cost function. Its extension to multiple sensors means that the cost function must be generalized to include the effects of sensor interactions, such as duplicate coverage in certain cases. The generalization is interesting and can lead to the determination of optimal transition probabilities for multiple collaborating sensors in a network.
- **Privacy protection for mobility traces.** We showed that when anonymous mobility traces are published even with reduced spatial and/or temporal precision, the complete path history of participants may still be revealed by an adversary armed with a small amount of side information. An interesting next problem is to devise techniques for privacy protection of the traces without compromising knowledge that is intended to be learned from them. This will require a delicate balance between preserving the essential properties of traces for a particular objective and fuzzing the details to avoid leak of sensitive information.

LIST OF REFERENCES

LIST OF REFERENCES

- [1] NASA's Ames Research Center. NASA Ames scientist develops cell phone chemical sensor. http://www.nasa.gov/centers/ames/news/features/2009/cell_phone_sensors.html.
- [2] Dorit S. Hochbaum and Barak Fishbain. Nuclear Threat Detection with Mobile Distributed Sensor Networks. *Annals of Operations Research*, 2009.
- [3] Prashanth Mohan, Venkat Padmanabhan, and Ramachandran Ramjee. Nericell: Rich Monitoring of Road and Traffic Conditions Using Mobile Smartphones. In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Raleigh, NC, USA, November 2008.
- [4] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring. In *Proceedings of the ACM International Conference on Mobile Systems, Applications and Services (MobiSys)*, Breckenridge, CO, USA, June 2008.
- [5] Santpal Singh Dhillon and Krishnendu Chakrabarty. Sensor Placement for Effective Coverage and Surveillance in Distributed Sensor Networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, New Orleans, LA, USA, March 2003.
- [6] Ronald W. Lee and James J. Kulesz. A Risk-Based Sensor Placement Methodology. Technical report, Computational Sciences and Engineering Division, Oak Ridge National Laboratory, 2006.
- [7] Ashok Sundaresan, Pramod K. Varshney, and Nageswara S. V. Rao. Distributed Detection of a Nuclear Radioactive Source Using Fusion of Correlated Decisions. In *Proceedings of the International Conference on Information Fusion*, Quebec City, Quebec, Canada, July 2007.
- [8] Thomas Karagiannis, Jean-Yves L. Boudec, and Milan Vojnovic. Power Law and Exponential Decay of Inter Contact Times between Mobile Devices. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, Montreal, Quebec, Canada, September 2007.
- [9] Dartmouth College. CRAWDAD. <http://crawdad.cs.dartmouth.edu/>.
- [10] ShangHai Grid. <http://www.cse.ust.hk/dcrg>.
- [11] Chris Y. T. Ma, David K. Y. Yau, Jren chit Chin, Nageswara S. V. Rao, and Mallikarjun Shankar. Matching and Fairness in Threat-Based Mobile Sensor Coverage. *IEEE Transactions on Mobile Computing*, 8(12):1649–1662, 2009.

- [12] Chris Y. T. Ma, David K. Y. Yau, Nung Kwan Yip, Nageswara S. V. Rao, and Jiming Chen. Stochastic Steepest-Descent Optimization of Multiple-Objective Mobile Sensor Coverage. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*, Genoa, Italy, June 2010.
- [13] Mihaela Cardei, My T. Thai, Yingshu Li, and Weili Wu. Energy-Efficient Target Coverage in Wireless Sensor Networks. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, Miami, FL, USA, March 2005.
- [14] Xiaorui Wang, Guoliang Xing, Yuanfang Zhang, Chenyang Lu, Robert Pless, and Christopher Gill. Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks. In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angeles, CA, USA, November 2003.
- [15] Thomas Clouqueur, Veradej Phipatanasuphorn, Parameswaran Ramanathan, and Kewal K. Saluja. Sensor Deployment Strategy for Target Detection. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, Atlanta, GA, USA, September 2002.
- [16] Maxim A. Batalin, Gaurav S. Sukhatme, Yan Yu, Richard Pon, Jason Gordon, Mohammad H. Rahimi, William J. Kaiser, Gregory J. Pottie, and Deborah E. Estrin. Task Allocation for Event-Aware Spatiotemporal Sampling of Environmental Variables. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Alberta, Canada, August 2005.
- [17] Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage Control for Mobile Sensing Networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.
- [18] Guiling Wang, Guohong Cao, and Thomas F. La Porta. Movement-Assisted Sensor Deployment. *IEEE Transactions on Mobile Computing*, 5(6):640–652, 2006.
- [19] Seapahn Meguerdichian, Farinaz Koushanfar, Miodrag Potkonjak, and Mani B. Srivastava. Coverage Problems in Wireless Ad-hoc Sensor Networks. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, Anchorage, AK, USA, April 2001.
- [20] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu, and Miodrag Potkonjak. Exposure in Wireless Ad-hoc Sensor Networks. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, Rome, Italy, July 2001.
- [21] Seapahn Megerian, Farinaz Koushanfar, Gang Qu, Giacomino Veltri, and Miodrag Potkonjak. Exposure in Wireless Sensor Networks: Theory and Practical Solutions. *Wireless Network*, 8(5):443–454, 2002.
- [22] Nageswara S. V. Rao, Nachimuthu Manickam, and Vladimir Protopopescu. Connectivity of Visibility Graphs and Terrain Model Acquisition by Robot Teams. *Congressus Numerantium*, 120:129–137, 1996.

- [23] Nageswara S. V. Rao, Vladimir Protopopescu, and Nachimuthu Manickam. Cooperative Terrain Model Acquisition by a Team of Two or Three Point-Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Minneapolis, MN, April 1996.
- [24] Nageswara S. V. Rao. Terrain Model Acquisition by Mobile Robot Teams and n-Connectivity. *Distributed Autonomous Robotic Systems 4*, 2000.
- [25] Nabhendra Bisnik, Alhussein Abouzeid, and Volkan Isler. Stochastic Event Capture Using Mobile Sensors Subject to a Quality Metric. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, Los Angeles, CA, USA, September 2006.
- [26] Wei-Jen Hsu, Kashyap Merchant, Haw-Wei Shu, Chih-Hsin Hsu, and Ahmed Helmy. Weighted Waypoint Mobility Model and its Impact on Ad-hoc Networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(1):59–63, 2005.
- [27] David B. Johnson and David A. Maltz. Dynamic Source Routing in Ad-hoc Wireless Networks. *Mobile Computing*, 5:153–181, 1996.
- [28] Carl D. Meyer. The Role of the Group Generalized Inverse in the Theory of Finite Markov Chains. *SIAM Review*, 17:443–464, 1975.
- [29] Paul J. Schweitzer. Perturbation Theory and Finite Markov Chains. *Journal of Applied Probability*, 5:401–413, 1968.
- [30] Bruce Hajek. Cooling Schedules for Optimal Annealing. *Mathematics of Operations Research*, 13(2):311–329, 1988.
- [31] David K. Y. Yau, Nung Kwan Yip, Chris Y. T. Ma, Nageswara S. V. Rao, and Mallikarjun Shankar. Quality of Monitoring of Stochastic Events by Periodic and Proportional-Share Scheduling of Sensor Coverage. In *Proceedings of the ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, Madrid, Spain, December 2008.
- [32] Chris Y. T. Ma, David K. Y. Yau, Nung Kwan Yip, Nageswara S. V. Rao, and Jiming Chen. Performance Analysis of Stochastic Network Coverage with Limited Mobility. In *Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Macau, China, October 2009.
- [33] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek Abdelzaher. Range-Free Localization Schemes for Large Scale Sensor Networks. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, San Diego, CA, USA, September 2003.
- [34] E. B. Cochran. New Concepts of the Learning Curve. *The Journal of Industrial Engineering*, 11(4):317–327, 1960.
- [35] Guanghui He and Jennifer C. Hou. Tracking Targets with Quality in Wireless Sensor Networks. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, Washington, DC, USA, November 2005.
- [36] Ralph E. Lapp and Howard L. Andrews. *Nuclear Radiation Physics*. Prentice Hall, 1948.

- [37] Mark E. Crovella and Azer Bestavros. Self-Similarity in World Wide Web traffic: Evidence and Possible Causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846, 1997.
- [38] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson. On the Self-Similar Nature of Ethernet Traffic. In *Proceedings of the ACM Conference of the Special Interest Group on Data Communication (SIGCOMM)*, San Francisco, CA, USA, September 1993.
- [39] Vijay Gupta, David E. Jeffcoat, and Richard M. Murray. On Sensor Coverage by Mobile Sensors. In *Proceedings of the IEEE Conference on Decision and Control*, San Diego, CA, USA, December 2006.
- [40] Sheldon M. Ross. *Stochastic Processes*. Wiley, New York, 1996.
- [41] Chris Y. T. Ma, David K. Y. Yau, Nung Kwan Yip, and Nageswara S. V. Rao. Privacy Vulnerability of Published Anonymous Mobility Traces. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, Chicago, IL, USA, September 2010.
- [42] Latanya Sweeney. k-Anonymity: A Model for Protecting Privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
- [43] Roberto J. Bayardo and Rakesh Agrawal. Data Privacy through Optimal k-Anonymization. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, Tokyo, Japan, April 2005.
- [44] Xiaokui Xiao and Yufei Tao. m-Invariance: Towards Privacy Preserving Replication of Dynamic Datasets. In *Proceedings of the ACM Special Interest Group on Management of Data Conference (SIGMOD)*, Beijing, China, June 2007.
- [45] David Martin, Daniel Kifer, Ashwin Machanavajjhala, Johannes Gehrke, and Joseph Y. Halpern. Worst-Case Background Knowledge for Privacy-Preserving Data Publishing. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, Istanbul, Turkey, February 2007.
- [46] Mehmet Ercan Nergiz, Maurizio Atzori, Yucel Saygin, and Baris Guç. Towards Trajectory Anonymization: a Generalization-Based Approach. *Transactions on Data Privacy*, 2(1), 2009.
- [47] Osman Abul, Francesco Bonchi, and Mirco Nanni. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Database. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, Cancun, Mexico, April 2008.
- [48] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-Preserving Data Mining. *ACM SIGMOD Record*, 29(2):439–450, 2000.
- [49] Fosca Giannotti and Dino Pedreschi. *Mobility, Data Mining and Privacy: Geographic Knowledge Discovery*. Springer Publishing Company, Incorporated, 2008.
- [50] Arvind Narayanan and Vitaly Shmatikov. Robust De-anonymization of Large Sparse Datasets. In *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, USA, May 2008.

- [51] Marco Gruteser and Dirk Grunwald. Anonymous Usage of Location-Based Services through Spatial and Temporal Cloaking. In *Proceedings of the ACM International Conference on Mobile Systems, Applications and Services (MobiSys)*, San Francisco, CA, USA, May 2003.
- [52] Bugra Gedik and Ling Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*, Columbus, OH, USA, June 2005.
- [53] Baik Hoh, Marco Gruteser, Hui Xiong, and Ansaf Alrabady. Preserving Privacy in GPS Traces via Uncertainty-Aware Path Cloaking. In *Proceedings of the ACM Computer and Communications Security Conference (CCS)*, Alexandria, VA, USA, October 2007.
- [54] Joseph Meyerowitz and Romit Roy Choudhury. Hiding Stars with Fireworks: Location Privacy through Camouflage. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, Beijing, China, September 2009.
- [55] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24), February 2009. <http://crawdad.cs.dartmouth.edu/epfl/mobility>.
- [56] Karl Pearson. *Royal Society Proceedings*, 1895.
- [57] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

VITA

VITA

Yu Tak Ma was born in Hong Kong in 1982. He received his Bachelor of Engineering with First Class Honours in Computer Engineering from the Department of Computer Science and Engineering, the Chinese University of Hong Kong, Hong Kong in 2004; and Master of Philosophy in Computer Science and Engineering from the Department of Computer Science and Engineering, the Chinese University of Hong Kong, Hong Kong in 2006. Yu Tak Ma entered Purdue University in the Fall of 2006, and received his Doctor of Philosophy degree in December 2010, under the direction of Prof. David K. Y. Yau. During his time at Purdue University, he was awarded a Bilsland Dissertation Fellowship, a Purdue University Graduate Student Award for Outstanding Teaching, and a Purdue Summer Research Grant. His research interests are in the performance and privacy study of wireless networks and mobile sensor networks.