UC Irvine

UC Irvine Electronic Theses and Dissertations

Title

Distributed Strategy Selection Over Graphs: Optimality and Privacy

Permalink

https://escholarship.org/uc/item/8kf1h60c

Author

Rezazadeh, Navid

Publication Date

2022

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA,
IRVINE

Distributed Strategy Selection Over Graphs: Optimality and Privacy

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Mechanical and Aerospace Engineering

by

Navid Rezazadeh

Dissertation Committee:
Professor Solmaz Kia, Chair
Professor Haithem Taha
Professor Tryphon Georgiou

2022

© 2022 Navid Rezazadeh

DEDICATION

To Mom and Dad...

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Page

LIST OF ALGORITHMS

ACKNOWLEDGEMENTS

This thesis is the result of multiple people's hard work and support, for which I am grateful. First and foremost, I want to express my gratitude to Professor Solmaz Kia, my Ph.D. advisor, for his encouragement and direction during this process. Her unwavering support has allowed me to freely pursue my research interests. Her vision have motivated me to go beyond my comfort zone and strive to be a better version of myself every day.

Professor Tryphon Georgiou and Professor Haithem Taha, members of my Ph.D. committee, are to be thanked for taking time out of their busy schedules to serve on the committee. Their presence and commitment as mechanical and aerospace engineering department faculty has always ensured and determined an ever-evolving and supportive atmosphere.

I would like to thank the generous funding provided by National Science Foundation (NSF) for my work. This work is supported by NSF CAREER award ECCS-1653838 and IIS-SAS-1724331.

For their help and support, I'd like to thank Noah Esteki, Changwei Chen, Hossein Moradian, Donipolo Ghimire, Yi-Fan Chung, Minwon Seo, Robert Palfini, and Jianan Zhu from KCS-lab. Working with them in KCS-lab together over the past five years has been a great privilege.

I would want to express my gratitude to my mom and dad for their unconditional love, patience, and sacrifice during this journey. They were always there to encourage me whenever I faced difficulties. Without their support, I would be unable to complete my thesis work.

CURRICULUM VITAE

Navid Rezazadeh

EDUCATION

Doctor of Philosophy in Mechanical and Aerospace Engineering **2022**
University of California, Irvine *Irvine, California*

Master of Science in Mechanical and Aerospace Engineering **2017**
University of California, Irvine *Irvine, California*

Bachelor of Science in Mechanical Engineering **2013**
Sharif University of Technology *Tehran, Iran*

RESEARCH EXPERIENCE

Graduate Research Assistant **2016–2022**
University of California, Irvine *Irvine, California*

Research Assistant **2014–2016**
Missouri University of Science and Technology Rolla, Missouri

MANUSCRIPTS IN PREPARATION

[J5] **Navid Rezazadeh** and Solmaz S. Kia, "Distributed submodular maximization: trading performance for privacy"

JOURNAL PUBLICATIONS

[J4] **Navid Rezazadeh** and Solmaz S. Kia, "A study of privacy preservation in average consensus algorithm via deterministic obfuscation signals", *IEEE Transactions on Control of Network Systems*, submitted, 2022.

[J3] **Navid Rezazadeh**, Maxwell Kolarich, Solmaz S. Kia and Negar Mehr, "Learning Contraction Policies from Offline Data", *IEEE Robotics and Automation Letters*, 2022.

[J2] **Navid Rezazadeh** and Solmaz S. Kia, "Distributed Strategy Selection: A Submodular Set Function Maximization Approach ", *Automatica*, 2021.

[J1] **Navid Rezazadeh** and Solmaz S. Kia, "A sub-modular receding horizon solution for mobile multi-agent persistent monitoring", *Automatica*, 2019.

CONFERENCE PUBLICATIONS

[C4] **Navid Rezazadeh**, Maxwell Kolarich, Solmaz S. Kia and Negar Mehr, "Learning Contraction Policies from Offline Data", *IEEE International Conference on Robotics and Automation*, 2022.

[C3] **Navid Rezazadeh** and Solmaz S. Kia, "Multi-agent maximization of a monotone submodular function via maximum consensus", *IEEE Conference on Decision and Control* , 2021.

[C2] **Navid Rezazadeh** and Solmaz S. Kia, "A sub-modular receding horizon approach to persistent monitoring for a group of mobile agents over an urban area", *IFAC Workshop on Distributed Estimation and Control in Networked Systems*, 2019.

[C1] **Navid Rezazadeh** and Solmaz S. Kia, "Privacy preservation in a continuous-time static average consensus algorithm over directed graphs", *American Control Conference*, 2018.

TECHNICAL TALKS

NeurIPS Safe and Robust Control of Uncertain Systems	**2021**
Southern California Control Workshop, UC Irvine	**2021**
IFAC Workshop on Distributed Estimation and Control in Networked Systems	**2019**
Southern California Control Workshop, UC Riverside	**2018**
American Control Conference	**2018**

TEACHING EXPERIENCE

Teaching Assistant, Dynamics **2022**
University of California, Irvine *Irvine, California*

Teaching Assistant, Robot Design **2021**
University of California, Irvine *Irvine, California*

Teaching Assistant, Capstone Project LiDAR **2021**
University of California, Irvine *Irvine, California*

Teaching Assistant, Vibrations **2020**
University of California, Irvine *Irvine, California*

Teaching Assistant, Robot Motion Planning and Navigation **2020**
University of California, Irvine *Irvine, California*

Teaching Assistant, Control Systems **2019**
University of California, Irvine *Irvine, California*

ABSTRACT OF THE DISSERTATION

Distributed Strategy Selection Over Graphs: Optimality and Privacy

By

Navid Rezazadeh

Doctor of Philosophy in Mechanical and Aerospace Engineering

University of California, Irvine, 2022

Professor Solmaz Kia, Chair

This dissertation contributes toward distributed strategy selection for networked mobile agents (robots, humans, unmanned aerial vehicles, etc.) for problems where the goal is to maximize the overal utility of the agents. The objective is the design of infrastructure-free decentralized cooperative decision-making algorithm that can have a robust performance in encountering uncertainties of the system. The main driving application of this work is cooperative strategy selection solutions for tasks such as distributed area patrolling, persistent monitoring, sensor placement, and route selection where avoiding action overlaps and maximizing the performance of the agents is challenging. This dissertation work is an effort to design a decentralized strategy selection algorithm relying on the local processing power of the agents and the communication network among them.

To motivate our work, We study the problem of persistent monitoring of the finite number of inter-connected geographical nodes for event detection via a group of heterogeneous mobile agents. We tie a utility function to the maximum reward available in each point of interest and use it in our strategy selection algorithm to incentivize the agents to visit the geographical nodes with higher rewards. We show that the design of an optimal monitoring policy to maximize the gathered reward over a mission horizon is an NP-hard problem. By showing that the reward function is a monotone increasing and submodular set function, we

formulate a general utility maximization problem as maximizing a submodular set function subject to partition matroid. We then proceed to propose a suboptimal strategy selection algorithm with known optimality bound. We work in the value oracle model where the only access of the agents to the utility function is through a black box that returns the utility function value. The agents are communicating over a connected undirected graph and have access only to their own strategy set. Hence, our objective is to propose a polynomial-time distributed algorithm to obtain a suboptimal solution with guarantees on the optimality bound. Our proposed algorithm is based on a distributed stochastic gradient ascent scheme built on the multilinear-extension of the submodular set function. We use a maximum consensus protocol to minimize the inconsistency of the shared information over the network caused by a delay in the flow of information while solving for the fractional solution of the multilinear extension model. Furthermore, we propose a distributed framework for finding a set solution using the fractional solution. We show that our distributed algorithm results in a strategy set that when the team objective function is evaluated at the worst case the objective function value is in $1 - 1/\mathrm{e}$ of the optimal solution. However, our proposed communication protocol trivially informs adversarial elements on the selected strategies by the agents. Our next contribution is to design a distributed algorithm that enables each agent to find a suboptimal policy locally with a guaranteed level of privacy. We base our modified algorithm's privacy preservation characteristic on our proposed stochastic rounding method and tie the level of privacy to the variable $\gamma \in [0, 1]$. That is, the policy choice of an agent can be determined with the probability of at most γ. We show that our distributed algorithm results in a strategy set that when the team's objective function is evaluated in the worst case, the objective function value is in $1 - (1/\mathrm{e})^{h(\gamma)} - O(T)$ of the optimal solution, highlighting the interplay between level of optimality gap and guaranteed level of privacy. To address the problem of decreased performance bound as the result of increased privacy levels we explore other methods of privacy preservation. We particularly study the privacy preservation methods in consensus-based communication protocols. We study the problem

of privacy preservation of the continuous-time Laplacian static average consensus algorithm using additive perturbation signals. We consider this problem over a strongly connected and weight-balanced digraph. Starting from a local reference value, in static average consensus algorithm, each agent constantly communicates with its neighboring agents to update its local state to compute the average of the reference values across the network. Since every agent transmits its local reference value to its in-neighbors, the reference value of the agents is trivially disclosed. We investigate the possibility of preserving the privacy of the reference value of the agents by adding admissible perturbation signals to the local dynamics and the transmitted out signals of the agents. Admissible additive perturbation signals are those signals that do not perturb the final convergence point of the algorithm from the average of the reference values of the agents. Our results show that if an adversarial agent has access to the output of another agent and all the input signals transmitted to that agent, the adversary can discover the private reference value of that agent, regardless of the perturbation signals. Otherwise, the privacy of the agent can be preserved. Our proposed randomized distributed strategy selection is prone to requiring an excessive number of samples in special scenarios. To address this problem, we look into reusing offline samples and modeling the system through neural networks. However, neural networks are notorious to be unpredictable and therefore not suitable to be used in dynamic systems. We proposed a method of certifying neural networks in the context of dynamical systems and policy-making using contraction theory to address the problem of uncertainty.

Chapter 1

Introduction

1.1 Motivation

In extension of cities and technology there is always a need for surveillance to monitor for incidences of interest. Traditionally, the surveillance systems are stationary and usually cover limited areas. The cost and communication bandwidth limitation bounds the number of the stationary sensors that can be deployed. To solve the coverage within the limits of the system, use of mobile sensors, e.g. aerial sensors, which the infrastructure can move within the urban area is of interest. Hence, it is of profound importance to design a dispatch policy that orchestrates the topological distribution of a set of mobile sensors such that the best service for a global monitoring task is obtained with a reasonable computational cost. Long term multi-agent patrolling of an area offers a low cost and effective monitoring solution for applications such as discovering forest fires and oil spillage in their early stages, and locating endangered animals in a large habitat.

Closed-form optimal monitoring policies and strategies could exist when some constraints are place on the problem. However, operational uncertainties such as node/route/agent

appearance and removal are inevitable during a long term persistent monitoring. Based on the area where agents are deployed, dropping the constraints and modeling the context-related uncertainties can yield a more robust and reliable patrolling strategy while making the calculation of an optimal long term patrolling strategy intractable. Hence, one approach of solving these class of problems is finding sub-optimal strategies with known optimality gap. The problem becomes even more challenging when the agents are asked to cooperatively decide on their strategy through talking to each other over a communication graph. The cooperation is necessary to reduce the overlap of the agents' actions.

While decentralized solutions have been shown to be more efficient and resilient in the face of uncertainty, several issues are constantly at the forefront of this class of solutions. Because agent collaboration necessitates the transmission of information, adversary actors might tap into the communication network and coordinate attacks against the agents. As a result, in order to get agents to engage in a suggested distributed system, their privacy concerns must be adequately handled. This dissertation is an attempt to solve the problem of distributed strategy selection with a focus on privacy issues.

1.2 Literature Survey

In this section, we review the state-of-art distributed mobile sensor dispatch and networked strategy selection results in the literature. Then we move to reviewing the privacy preservation algorithms in the networked systems. Finally, we review some recent results on the use of certified deep learning methods in modeling and control of complex dynamic systems and later we discuss how it can be a start point for future work.

1.2.1 Distributed Monitoring

In recent years, coordinating the movement of mobile sensors to cover areas that have not been adequately sampled/observed has been explored in controls, wireless sensors and robotic communities with problems related to coverage, exploration, and deployment. Many of the proposed algorithms strive to spread sensors to desired positions to obtain a stationary configuration such that the coverage is optimized, see e.g.,[5, 6, 7, 8, 9, 10, 11, 12]. Some sensor placement problems such [8, 10, 11, 12] are context-aware, and include also a period of exploration and observation to increase the knowledge used to find the optimal residing position of the sensors. In this thesis, instead of aiming to achieve an improved stationary network configuration as the end result of the sensors' movement, our objective is to explore context-aware mobility strategies that dynamically reposition the mobile sensors to maximize their utilization and contribution over a mission horizon. Motivating applications include persistent monitoring to discover forest fires [13] or oil spillage in its early stages [14], locating endangered animals in a large habitat [15] and event detection in urban environments [16]. Specifically, we consider a persistent monitoring of a set of finite $\mathcal{V}$ inter-connected geographical nodes via a set of finite $\mathcal{A}$ mobile sensors/agents, where $|\mathcal{V}| > |\mathcal{A}|$. The mobile agents are confined to a set of pre-specified edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, e.g., aerial or ground corridors, to traverse from one node to another, see Fig. 1.1. Depending on their vehicle type, agents may have to take different edges to go from one node to another. Also, they may have different travel times along the same edge. We study dispatch policy that orchestrates the topological distribution of the mobile agents such that an optimized service for a global monitoring task is provided with a reasonable computational cost. To quantify the service objective we assign to each node $v \in \mathcal{V}$ the reward function $R_v(t)$ composed of nonnegative concave and increasing function of time. For example, in data harvesting or health monitoring, the concave function can be weighted idle time of the node v or in event detection, it can be the probability of at least one event taking place at inter-visit times. Optimal patrolling

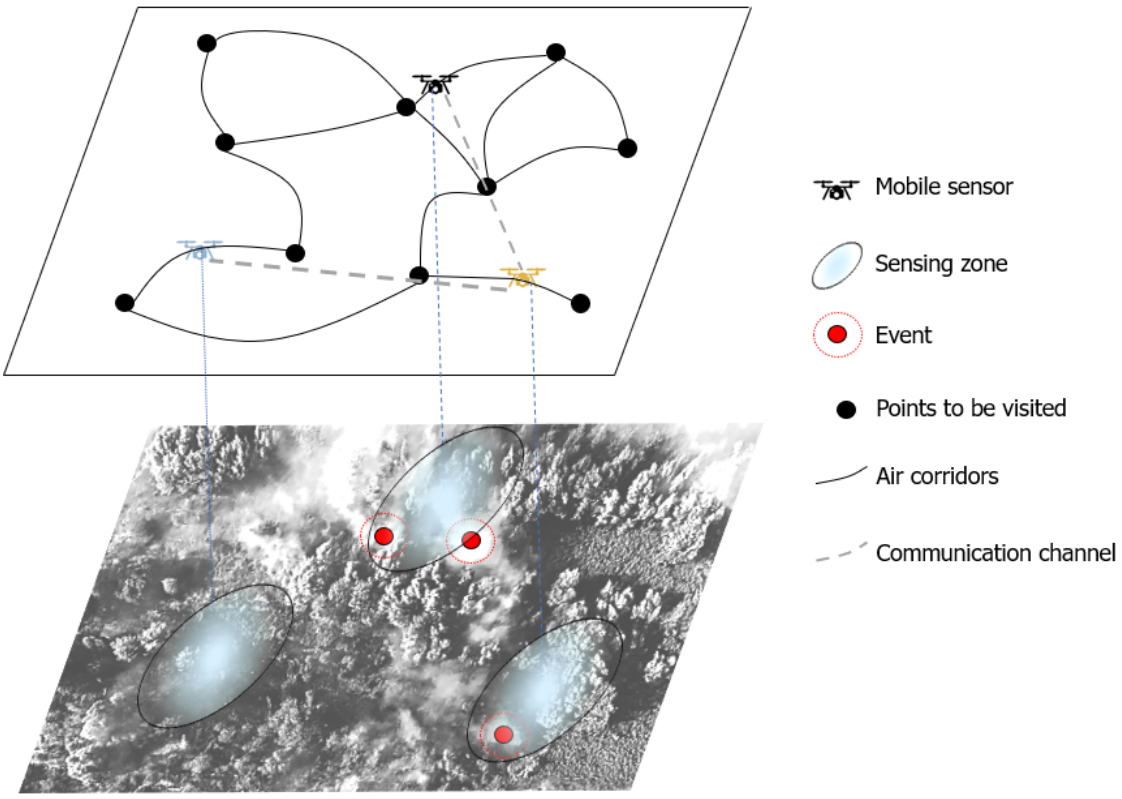


Figure 1.1 – Examples of a set of geographical nodes of interest and the edges between them spread in a forest with a few mobile sensors dispatched to detect the location of possible fires.

designs a dispatch policy (what sequence of nodes to visit at what times by which agents) to score the maximum collective reward for the team over the mission horizon. However, as we explain below, this problem is NP-hard. Our aim then is to design a suboptimal solution that has polynomial time complexity.

Dispatch policy design for patrolling/monitoring of geographical nodes can be divided into two categories: the edges to travel between the nodes are not specified (design in continuous edge space) or otherwise (design in discrete edge space). When there are no prespecified inter-node edges, the optimal patrolling policy design includes also finding the optimal inter-node trajectories that the agents should follow without violating their mobility limits. In some applications, however, the mobile agents are confined to travel through pre-specified known edges between the nodes. For example, in a smart city setting, regulations can

restrict the admissible routes between the geographical nodes. In the dispatch policy design in discrete edge space, the complexity of finding the optimal policy for a single patrolling agent is the same as the complexity of solving the Traveling Salesman problem, where the computational complexity grows exponentially with the number of the nodes [17]. In case of multiple patrolling agents, the problem is even more complex, since each agent's policy design depends on the other agents' policy. This problem is formalized in earlier studies such as [18, 19]. Generally, when there are multiple edges to travel between every two nodes or when each node is connected to multiple other nodes, finding an optimal long term patrolling scheme is not tractable. Constraining the agents to travel through specific edges to traverse among the geographical nodes allows seeking optimal solutions for the problem. For example, when the connection topology between the geographical nodes is a path or a cyclic graph, optimal solutions for the problem are proposed in [20, 21, 22, 23]. To overcome the complexity issue on generic graphs, [24] explores forming different cycles in the graph and assigning agents to these cycles to patrol the nodes periodically and seeks to minimize the time that a node stays un-visited. Alternatively, [25] proposes agents to move to the most rewarding neighboring node based on their current location.

We propose a robust and suboptimal solution to the long term patrolling problem that we stated earlier. Instead of using the customary idle time, t, as a reward function, which reduces the optimal dispatch policy design to the minimum latency problem [26], we consider reward functions described by an increasing concave function. This allows modeling a wider class of patrolling problems such as patrolling for event detection. We let the utility function to be the sum of the rewards collected over the mission horizon by the mobile agents. We discuss that the design of optimal patrolling policy to maximize this utility over the mission horizon is an NP-hard problem. Specifically, we show that the complexity of finding the optimal policy increases exponentially with the mission horizon and number of agents. Next, we show that the utility function is a monotone increasing and submodular set function. To establish this result, we develop a set of auxiliary lemmas based on the Karamata's inequality [27].

Given the submodularity of the utility function, we propose a receding horizon sequential greedy algorithm to compute a suboptimal dispatch policy with a polynomial computation cost and guaranteed bound on optimality. The receding horizon nature of our solution induces robustness to uncertainties of the environment. Our next contribution is to add a new term to our utility function to compensate for the shortsightedness of the receding horizon approach, see Fig. 2.2. When agents patrol a large set of inter-connected nodes, this added term becomes useful by giving them an intuition of the existing reward in the farther nodes. In recent years, submodular optimization has been widely used in sensor and actuator placement problems [6, 7, 28, 29, 30, 31]. In comparison to the sensor/actuator placement problems, the challenge in our work is that the assigned policy per each mobile agent over the receding horizon is a dynamic scheduling problem rather than a static sensor placement. To deal with this challenge, we use the matroid constraint [32] approach to design our suboptimal submodular-based policy. Finally, we discuss how our algorithm can be implemented in a decentralized manner. A simulation study demonstrates our results. Our notation is standard, though to avoid confusion, certain concepts and notation are defined as the need arises.

1.2.2 Distributed Strategy Selection

Modern industries such as transportation, supply chain, energy, and finance are moving fast towards modular and distributed operations where communicating smart sub-systems are expected to coordinate their actions for the optimal operation of the entire system. Optimal strategy selection problems for these networked systems often appear as combinatorial optimization problems where the objective function is a submodular set function. Some example cases include sensor and actuator placement problems [28, 33], energy storage placement [34, 35], measurement scheduling [36], voltage control in smart grid [37], persistent monitoring via mobile robots [38]. For reasons such as robustness, scalability, privacy preser-

vation, and avoiding a single failure point, these optimal decision-making problems are highly desired to be solved in a distributed manner. While there has been a plethora of work in developing central solutions for submodular maximization, satisfactory distributed algorithmic solutions for in-network submodular maximization problems where agents communicate over a graph have remained elusive. in this thesis, we consider a distributed strategy selection problem that is modeled as submodular maximization subject to *partition matriod*. We seek a distributed solution in which the agents communicate over a connected undirected graph.

Submodular function maximization: A set function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ defined on the ground set $\mathcal{P}$ is *submodular* if $\forall\, \mathcal{S} \subset \mathcal{T} \subset \mathcal{P}$, and $p \in \mathcal{P} \setminus \mathcal{T}$ we have

$$f(\mathcal{S} \cup \{p\}) - f(\mathcal{S}) \geq f(\mathcal{T} \cup \{p\}) - f(\mathcal{T}). \tag{1.1}$$

Submodular set functions naturally possess the diminishing returns property, i.e., the gain of adding a particular element p to a set decreases or stays the same as the size of the set increases. Submodularity is an inherent property in many practical utility/objective functions such as weighted coverage functions, facility location service function, entropy, and mutual information functions, which appear in strategy selection problems such as sensor placement, measurement scheduling, workforce hiring, and database sampling [39]. Unlike minimization of submodular functions that can be done in polynomial time [40, 41], submodular function maximization problems are NP-hard [42]. Luckily, submodularity is a property of set functions with deep theoretical consequences that enables establishing constant factor approximate (suboptimal solutions) for submodular maximization problems. Research on problems involving the maximization of monotone submodular functions dates back to the work of Nemhauser, Wolsey, and Fisher in the 1970's [42, 43, 44]. A fundamental result by Nemhauser et al. [42] establishes that the simple *sequential greedy algorithm* is guaranteed to provide a constant 1/2-approximation factor solution for submodular maximization subject to matroid constraints. The sequential greedy algorithm reaches the final solution by

sequentially finding the best current decision based on the decisions made previously and without considering the consequences or interactions with future decisions. These bounds can be made tighter with additional knowledge on the diminishing return property of the submodular objective function quantified by *total curvature* $c \in [0, 1]$ [45]. For example [45] shows that the constant factor approximation for submodular maximization subject to a matroid constraint is $\frac{1}{1+c}$ [45].

More recently, another suboptimal solution for submodular maximization subject to matroid constraints with an improved optimality gap is proposed in the literature using the *multilinear* continuous relaxation of a submodular set function [4, 46, 47, 48, 49]. The relaxation transforms the discrete problem into a continuous optimization problem with linear constraints. Then, a continuous gradient-based optimization algorithm referred to as *continuous greedy algorithm*, is used to solve the continuous optimization problem. A suboptimal solution for submodular maximization subject to matroid constraint with the improved constant-factor approximation of $(1 - 1/\mathrm{e})$ then is obtained by proper *rounding* of the continuous-domain solution [4, 46]. This approach however requires a central authority to solve the problem. It is worth noting that the literature has shown that for monotone submodular functions, it is computationally hard to approximate this problem within a factor better than $1 - 1/e \approx 0.63\%$ [50].

Distributed submodular function maximization: In multi-agents setting, for example, multi-agent sensor placement problems where the agents are self-organizing autonomous mobile agents with communication and computation capabilities, it is desired to solve the strategy selection problems modeled as constrained submodular maximization problems in a distributed way without involving a central authority. The problems in distributed settings can be divided into two categories: distributed constraint problems and distributed utility problems. In a distributed constraint problem there is a shared utility but each agent has to choose its strategy from a local constraint set that is disjoint from other agents' and is only

known to the agent. An example is the heterogeneous coverage problem where each agent has a set of heterogeneous sensors while the area to cover is shared among them. The agents should decide what sensor(s) each to deploy to maximize the area coverage as a team. In distributed utility problems, however, the team's utility function is the sum of the separable local utilities and agents choose their strategies from a shared strategy set. An example case is the optimal Welfare problem [4] where each agent should make strategy choices from a joint set such that the sum of local utilities is maximized. Our focus in this thesis is on distributed solution design for a fragmented-constraint submodular maximization problem.

For distributed constraint problems, the sequential greedy algorithm can be implemented in a decentralized way through sequential message-passing or via sequential message-sharing through a cloud [38]. However, a decentralized sequential greedy algorithm comes with communication routing overhead. For agents communicating over a connected graph, implementing sequential message-passing requires finding the Hamiltonian path (a connected path that visits every agent on the graph only once) which is an NP-hard problem to solve. If Hamiltonian path does not exist in a graph, a path that visits the agents in least frequent times should be identified for communication efficient sequential message-passing. Moreover, it is shown that the order of sequence changes the actual approximation factor of the solution obtained by the sequential greedy algorithm [51]. The complexity of finding the sequence that delivers the best solution increases exponentially as the size and the connectivity of the communication network increases. Several attempts have been also undertaken to adapt the sequential greedy algorithm for large-scale submodular maximization problems by reducing the size of the problem through approximations [52] or using several processing units to achieve a faster sequential greedy algorithm, but with some sacrifices on the optimality bound [53, 54, 55, 56]. However, these decentralized implementations are mainly intended for parallel processing purposes and are not extendable to decentralized operations when agents communicate over connected graphs.

Some attempts have also been made in devising distributed solutions for submodular maximization using multi-linear extension approaches. For the class of distributed utility functions semi-distributed and fully distributed solutions with an optimality gap close to $(1-1/\mathrm{e})$ is studied in [57, 58, 59]. However, for the class of distributed constraint problems, the results in the literature are rather limited. For the special class of submodular set functions with curvature $c = 1$, and when each agent is limited to choose only a single strategy from its own strategy set, [2] has proposed an average consensus-based distributed algorithm to the maximization problem over connected graphs. The solution of [2] requires a closed-form expression of the multi-linear extension function. However, the computational complexity of constructing the closed-form of multi-linear extension of a submodular function and its derivatives increases exponentially with the size of the strategy set. Moreover, the result also depends on a centralized rounding scheme.

In this thesis, motivated by the improved optimality gap of the multilinear continuous relaxation-based algorithms, we develop a distributed implementation of the algorithm of [46] over a connected undirected graph. Particularly, we consider a distributed submodular set function maximization problem formulated by a shared utility function and disjoint strategy sets (fragmented-constraint class). Moreover, in our setup, the agents are allowed to choose multiple strategies from their strategy sets. We propose a gradient-based algorithm, which uses a maximum consensus scheme over the communication graph and results in a distributed implementation of the continuous greedy algorithm. The multi-linear extension function of a submodular function, as we review in Section 3.2, is equivalently the expected value of the submodular function evaluated at random sets obtained by picking strategies from the strategy set independently with a probability. This stochastic interpretation allows approximating the multi-linear extension function and its derivatives empirically with a reasonable computational cost via sampling from the strategy set [46]. Of course, as expected, this approach comes with a penalty on the optimality gap that is inversely proportional to the number of samples. In our algorithm, to manage the computational cost

of constructing the multilinear extension of the utility function and its derivatives, we use a sampling-based evaluation of the multilinear extension. Our careful analysis captures the effect of this approximation on our algorithm's optimally gap. We complete our solution by designing a distributed rounding procedure that computes the final suboptimal strategy of each agent. Rounding procedures, either deterministic or randomized, are widely used in the design and analysis of combinatorial optimization algorithms solved by continuous relaxation/approximation algorithms. Rounding procedures convert an optimal solution of a relaxed problem into an approximately optimal solution to the original problem. Pipage rounding [60] and randomized Pipage rounding [46] are examples of such procedures. However, these methods are designed for central submodular maximization solvers and do not trivially extend to the distributed settings. In a distributed setting, generally, the rounding procedure requires extra coordinating communication between the agents. However, our choice of maximum consensus algorithm as the agreements protocol between the agents removes the need for further communication. We show that after the coordination of the agents through maximum consensus for a given period of time, each agent can use a local randomized Pipage procedure to reach a deterministic set of strategies as its local solution without the necessity to interact with other agents. Furthermore, we show that the resulting global suboptimal strategy generated by our distributed algorithm lies in the feasible constraint set. Through rigorous analysis which takes into account the total curvature of the utility function, we show that our proposed distributed algorithm in finite time T achieves, with a known probability, a $\frac{1}{c}(1-\mathrm{e}^{-c})-O(1/T)$ optimality bound, where $1/T$ is the step size of the algorithm and the frequency at which agents communicate over the network. A numerical example demonstrates our results.

1.2.3 Private Strategy Selection

We consider a group of $\mathcal{A}$, $|\mathcal{A}| = N$ agents with communication and computation capabilities, interacting over a connected undirected graph $\mathcal{G}(\mathcal{A}, \mathcal{E})$. Each agent $a \in \mathcal{A}$ has a distinct discrete policy set $\mathcal{P}_a$ and wants to choose $\kappa_a \in \mathbb{Z}_{\geq 1}$ policies from its policy set such that a monotone increasing and submodular utility function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, $\mathcal{P} = \bigcup_{a \in \mathcal{A}} \mathcal{P}_a$, evaluated at all the agents' policy selection is maximized[1] While seeking a distributed solution for this, each agent wants to have a formal guarantee that its final policy choice stays private. Even though a distributed solution eliminates the necessity of information aggregation in a central location, inter-agent communication can still expose distributed network operations to adversarial eavesdroppers. These adversaries can be other agents in the network or outside eavesdroppers that intercept communication messages. Because in problem (4.1) the agents have joint utility function, privacy preservation is particularly a challenging problem. This problem falls in the so-called distributed-constraint submodular maximization class of problems in networked systems. In a distributed-constraint problem, there is a shared utility, but each agent has to choose its strategy from a local constraint set that is disjoint from other agents and is only known to the agent [2, 51, 61, 62, 63, 64]. An example is the heterogeneous coverage problem where each agent has a set of heterogeneous sensors while the area to cover is shared among them [39]. This is different than the distributed-utility problems such as Welfare problem [4] where the team's utility function is the sum of the separable local utilities, and agents choose their strategies from a shared strategy set [57, 58, 59].

Submodular maximization subject to matroid constraint is an NP-hard problem [42]. However, thanks to the inherent properties of submodular functions, suboptimal solutions with quantifiable approximation factors have been successfully proposed in the literature. The most well-known result is the *sequential greedy algorithm* that dates back to the 1970s by [42], guaranteeing 1/2-approximation factor solution for problem (4.1) when it is solved in a cen-

[1]For clarity, we provide a brief description of the notation and the definitions in Section **??**.

tralized manner. The sequential greedy algorithm preset a sequence, and each agent chooses its own best local policy given the choices of the preceding agents in the sequence. The sequential greedy algorithm can be implemented via sequential message-passing over a connected graph. However, this comes with routing overhead to identify the shortest path that visits every agent. But more importantly, in the sequential greedy algorithm, the agents' privacy is breached as each agent passes its local policy set to those proceeding it in the message-passing sequence.

More recently, another suboptimal solution for submodular maximization subject to matroid constraints with an improved approximation factor tor of $(1-1/\mathrm{e})$ is proposed [4, 46, 47, 48, 49]. This method relies on the use of continuous relaxation using a multilinear extension of submodular set functions and matroid polytope. The continuous relaxation is solved using a gradient ascent algorithm, and the integer solution then is rounded using appropriate rounding procedures such as Pipage rounding [60] or randomized Pipage rounding [46]. Besides its improved optimality gap, this approach has been shown to be amenable for distributed implementations that use synchronous inter-neighbor communication both for distributed-constraint problems [2, 61, 62] and distributed-utility problems [57, 58, 59]. However, none of the existing work in distributed submodular maximization addresses the privacy preservation of the agents formally. Our privacy preservation mechanism is different than existing perturbation methods for distributed algorithms such as differential privacy [65, 66], which rely on adding additive noises to the inter-agent communication massages. Differential privacy has also been the main approach in centralized submodular maximization problems subject to cardinality constraint [67, 68] and matroid constraint [69]. Instead of adding noises to data or inter-agent communications, the innovation in our work is to base our privacy preservation mechanism on our proposed stochastic rounding method and tie the level of privacy to a variable $\gamma \in [0, 1]$. γ defines the probability an agent's policy choice can be determined. We show that our distributed algorithm results in a strategy set that when the team's objective function is evaluated at worst case, the objective function value

is in $1-(1/\mathrm{e})^{1-\kappa_{\max}\sqrt{1-\gamma}}-O(T)$, $\kappa_{\max}=\max_{a\in\mathcal{A}}\kappa_a$ of the optimal solution in value oracle model, highlighting the interplay between level of optimality gap and guaranteed level of privacy.

1.2.4 Privacy of Networked Systems

Decentralized multi-agent cooperative operations have been emerging as effective solutions for some of today's important socio-economical challenges. However, in some areas involving sensitive data, for example in smart grid, banking or healthcare applications, the adaption of these solutions is hindered by concerns over the privacy preservation guarantees of the participating clients. Motivated by the demand for privacy preservation evaluations and design of privacy-preserving augmentations for existing decentralized solutions, in this work we consider the privacy preservation problem in the distributed static average consensus problem using additive obfuscation signals.

The static average consensus problem in a network of agents each endowed with a local static reference value consists of designing a distributed algorithm that enables each agent to asymptotically obtain the average of the static reference values across the network. The solutions to this problem have been used in various distributed computing, synchronization, and estimation problems as well as control of multi-agent cyber-physical systems. The average consensus problem has been studied extensively in the literature (see e.g., [70, 71, 72], [73]). The widely adopted distributed solution for the static average consensus problem is the simple first-order Laplacian algorithm in which each agent initializes its local dynamics with its local reference value and transmits this local value to its neighboring agents. Therefore, the reference value is readily revealed to the outside world, and thus the privacy of the agents implementing this algorithm is trivially breached. This work studies the multi-agent static average consensus problem under the privacy preservation requirement against internal and external passive eavesdroppers in the network. By passive, we mean agents that only listen

to the communication messages and want to obtain the reference value of the other agents without interrupting the distributed operation. The solution we examine is to induce privacy preservation property by adding obfuscation signals to the internal dynamics and the transmitted output of the agents.

Privacy preservation solutions for the average consensus problem have been investigated in the literature mainly in the context of discrete-time consensus algorithms over connected undirected graphs. The general idea is to add obfuscation signals to the transmitted out signal of the agents. For example, in one of the early privacy-preserving schemes, Kefayati, Talebi, and Khalaj [74] proposed that each agent adds a random number generated by zero-mean Gaussian processes to its initial condition. This way the reference value of the agents is guaranteed to stay private but the algorithm does not necessarily converge to the anticipated value. Similarly, in recent years, Nozari, Tallapragada and Cortes [75] also relied on adding zero-mean noises to protect the privacy of the agents. However, they develop their noises according to a framework defined based on the concept of differential privacy, which is initially developed in the data science literature [76, 77, 78, 79]. In this framework, [75] characterizes the convergence degradation and proposes an optimal noise in order to keep a level of privacy to the agents while minimizing the rate of convergence deterioration. To eliminate deviation from desired convergence point, Manitara and Hadjicostis [80] proposed to add a zero-sum finite sequence of noises to the transmitted signal of each agent, and Mo and Murray [3] proposed to add zero-sum infinite sequences. Because of the zero-sum condition on the obfuscation signals, however, [80] and [3] show that the privacy of an agent can only be preserved when the eavesdropper does not have access to at least one of the signals transmitted to that agent. Additive noises have also been used as a privacy preservation mechanism in other distributed algorithms such as distributed optimization [65] and distributed estimation [81, 82]. A thorough review of these results can be found in a recent tutorial work [83]. For the discrete-time average consensus, on a different approach, [84] uses a cryptographic approach to preserve the privacy of the agents. Moreover, [85] proposes to

use the dynamic average consensus algorithm of [86] as a privacy-preserving algorithm for the average consensus problem.

We consider the problem of privacy preservation of the continuous-time static Laplacian average consensus algorithm over strongly connected and weight-balanced digraphs using additive obfuscation signals. Similar to the reviewed literature above, in our privacy preservation analysis, we consider the extreme case that the eavesdroppers know the graph topology. But, instead of stochastic obfuscations, here we use deterministic obfuscations signals. These obfuscations are in the form of continuous-time integrable signals that we add to the transmitted out signal of the agents are also to the agreement dynamics of the agents. We refer to the obfuscation signals that do not disturb the final convergence point of the algorithm as *admissible obfuscation signals*. In our approach, instead of using by the customary zero-sum vanishing additive admissible signals, we start by carefully examining the stability and convergence proprieties of the static average consensus algorithm in the presence of the obfuscations to find the necessary and sufficient conditions on the admissible obfuscation signals. The motivation is to explore whether there exist other types of admissible obfuscation signals that can extend the privacy preservation guarantees. An interesting theoretical finding of our study is that the admissible obfuscation signals do not have to be vanishing. Also, we show that the necessary and sufficient conditions that specify the admissible obfuscation signals of the agents are highly coupled. We discuss how the agents can choose their admissible obfuscation signals locally with or without coordination among themselves. The conditions we obtain to define the locally chosen admissible obfuscation signals are coupled through a set of under-determined linear algebraic constraints with constant scalar free variables.

Understanding the nature of the admissible obfuscation signals is crucial in the privacy preservation evaluations, as it is rational to assume that the eavesdroppers are aware of the necessary conditions on such signals and use them to breach the privacy of the agents. In

our study, we evaluate the privacy preservation of the Laplacian average consensus algorithm with additive obfuscation signals against internal and external eavesdroppers, depending on whether the coupling variables of the necessary conditions defining the locally chosen admissible obfuscation signals are known to the eavesdropper or not. This way, we study privacy preservation against the most informed eavesdroppers and also explore what kind of guarantees we can provide against less informed eavesdroppers that do not know some parameters. We show that when the coupling variables are known to the eavesdroppers, they can use this extra piece of information to enhance their knowledge set to discover the private value of the other agents. In this case, our main result states that the necessary and sufficient condition for an eavesdropper to be able to identify the initial value of another agent is to have direct access to all the signals transmitted to and out of the agent. When this condition is not satisfied, the privacy guarantee is that the eavesdropper not only cannot obtain the exact reference value but also cannot establish an estimate on it. Precisely, to show that any agent i is private, we show that across the network there are arbitrarily different reference values, including for agent i, for which the signals received by the eavesdropper is exactly the same as those corresponding to the initializing the algorithm at the actual reference values. This shows that the use of deterministic obfuscation signals results in a stronger privacy guarantee than the stochastic approaches such as ϵ-differential privacy [75] and of [3] where even though the exact reference value is concealed, an estimate on the reference value can be obtained, see, e.g., [3, Fig. 4].

Our next contribution is to design asymptotic observers that internal and external eavesdroppers that have access to all the input and output signals of an agent can use to identify that agent's initial condition. For these observers, we also characterize the time history of their estimation error. Our results show that external eavesdroppers need to use an observer with a higher numerical complexity to compensate for the local state information that internal eavesdroppers can use. As another contribution, we identify examples of graph topologies in which the privacy of all the agents is preserved using additive admissible obfuscation sig-

nals. On the other hand, if the coupling variables of the necessary conditions defining the locally chosen admissible obfuscation signals are unknown to the eavesdroppers, we show that the eavesdroppers cannot reconstruct the private reference value of the other agents even if they have full access to all the transmitted input and output signals of an agent. We use input-to-state stability (ISS) results (see [87, 88]) to perform our analysis.

A preliminary version of our work has appeared in [89]. In this work the results are extended in the following directions: (a) we derive the necessary and sufficient conditions to characterize the admissible signals; (b) we study privacy preservation also with respect to external eavesdroppers; (c) we consider a general class of a set of measurable essentially bounded obfuscation signals; (d) we improve our main result from sufficient condition to necessary and sufficient condition.

1.2.5 Certified Neural Networks

While learning-based controllers have achieved significant success, they still lack safety guarantees. For instance, in general, the temporal evolution of a robot's trajectories under a learned policy cannot be certified. On the other hand, when a system's dynamics are known, control-theoretic properties, such as stability and contraction, directly examine the temporal progression of a system's states to verify whether a system remains within a safe set, and whether the system's trajectories converge. In this work, we seek to enforce the desired temporal evolution of the closed-loop system's states while learning the policy from an offline set of data, i.e. we seek to learn control policies such that under the learned policy, the convergence of a robot's trajectories is achieved.

To achieve such trajectory convergence, our design approach leverages Contraction theory [90]. Contraction theory provides a framework for identifying the class of nonlinear dynamic systems that have asymptotic convergent trajectories. Intuitively, a region of the

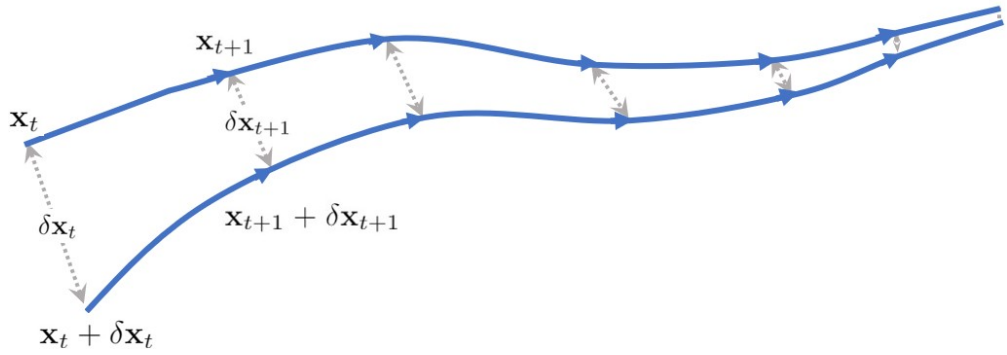


Figure 1.2 – The schematic of two neighboring trajectories that exhibit contraction. The distance between the trajectories decreases over time: $\|\delta\mathbf{x}_{t+1}\| < \|\delta\mathbf{x}_t\|$, i.e. trajectories converge.

state space is a contraction space if the distance between any two close neighboring trajectories decays over time. This notion of convergence is relevant to many robotic tasks such as tracking controllers where we want a robot to either reach a goal or track a reference trajectory. In this work, we want to learn policies from offline data such that they achieve convergence of a robot's trajectories in closed loop. While contraction theory provides a simple and intuitive characterization of convergent trajectories, finding the distance metric with respect to which a robot's trajectories exhibit contraction – which is called the *contraction metric* – is often non-trivial. To address this challenge, we propose to *jointly* learn the robot policy and its corresponding contraction metric.

We learn the robot dynamics model from an offline data set consisting of the robot's state and input trajectories. This setting is similar to the setting of offline model-based reinforcement learning (RL) where a dynamics model and a policy are learned from a set of robot trajectories that are collected offline. Learning from offline data is appropriate for safety-critical applications where online data collection is dangerous [91]. We learn a dynamics model of the system from the data and propose a data augmentation algorithm for learning contraction policies. Randomly sampled states are propagated forward in time through the learned dynamics model to generate auxiliary sample trajectories. We then learn both our policy and our contraction metric such that the distance between the robot trajectories from the data set and the auxiliary sample trajectories decreases over time. Learning contraction

policies is particularly relevant to offline RL as it allows us to regard the errors in the learned dynamics model as external disturbances and obtain a tracking error bound in regions where the learning errors of the dynamics model are bounded [92, 93].

We evaluate the performance of our proposed framework on a set of simulated robotic goal-reaching tasks. The performance of our proposed framework is compared with a number of control algorithms. We demonstrate that as a result of enforcing contraction, the robot's trajectories converge faster to the goal position with a higher degree of accuracy. It is further shown that learning contraction policies increases the robustness of the learned policy with respect to learned dynamics model mismatch, i.e. enforcing contraction increases the robustness of the learned policies. In summary, our contributions are the following:

- We propose a framework for learning convergent robot policies from an offline data set using Contraction theory.
- We develop a data augmentation algorithm for learning contraction policies from the offline data set.
- We provide a formal analysis for bounding contraction policy performance as a function of dynamics model mismatch.
- We perform numerical evaluations of our proposed policy learning framework and demonstrate that enforcing contraction results in favorable convergence and robustness performance.

For systems with unknown dynamics, several offline RL algorithms have been developed recently which either directly learn a policy using an offline data-set [94, 95, 96, 97] or learn a surrogate dynamics model from the offline data to learn an appropriate policy [98, 99]. However, the majority of such RL algorithms lack formal safety guarantees, and the convergent behavior of the learned policies is not certified [100, 101].

When the system dynamics are known, robust and certifiable control policy design can be achieved through various control-theoretic methods such as reachability analysis [102], Funnels [103, 104], and Hamilton-Jacobi analysis [105, 106]. Lyapunov stability criteria, Contraction Theory, and Control Barrier Functions have been extensively utilized for providing strong convergence guarantees for nonlinear dynamical systems [90, 107, 108, 109, 110]. However, even when the dynamics are known, finding a proper Lyapunov function or a control barrier function is itself a challenging task. To address these challenges, learning algorithms have been utilized for learning the Lyapunov and Control Barrier Functions [111, 112, 113]. [100] and [114] propose to learn contraction metrics to find contraction control policies for known systems.

Various recent works have considered combining control-theoretic tools with learning algorithms to enable learning safe policies even when dynamics are unknown. [115, 116] consider learning stable dynamics models. In [117], Contraction theory is used to learn stabilizable dynamics models of unknown systems. In [101, 118], Lyapunov functions are used for ensuring the stability of the learned policies. [119] proposed to learn the system dynamics and its corresponding Lyapunov function jointly to ensure the stability of the learned dynamics model.

In this work, we consider learning contraction policies from offline data for systems with unknown dynamics. Our work is closely related to [100] and [114], where Contraction theory has been used for certifying convergent trajectories. The current work is different in that, unlike these approaches where dynamics are explicitly known and assumed to be control-affine, we consider access to only an offline data set. We assume that we can learn an implicit model of system dynamics, in the form of a neural network function approximator, and provide robustness guarantees with respect to the errors of the learned dynamics model.

1.3 Objective

The objective of this dissertation work is to develop an effective infrastructure-free UWB-based cooperative localization. The main driving application of this work is cooperative localization solutions for firefighters and first-responders in extreme indoor environments where taking inter-agent measurements in line-of-sight (LoS) and maintaining network connectivity are challenging. This dissertation work is end-to-end, spanning algorithm design, theoretical modeling and analysis, testbed development, and experimental demonstrations/validation.

In cooperative localization, taking account of the correlations between agents is important for the consistency of the algorithm. However, the exact account of cross-correlation terms comes with high computation and communication cost, a stringent requirement on network connectivity, and communication channel condensations. In an effort to develop practical cooperative localization algorithms under limited connectivity, in this dissertation, we investigate loosely coupled algorithms where the correlations are accounted for in an implicit manner by a conservative but consistent estimate of the joint covariance matrix of the team members. We also study the design of a server-assisted loosely-coupled cooperative localization is investigated such that the cooperative localization system can inherit the advantages of both loosely coupled cooperative localization (high flexibility and low cost) and tightly coupled cooperative localization (high accuracy).

Another objective of this work is the accurate modeling. We also aim to address the challenges in the proper processing of the UWB range measurements in the framework of the our proposed cooperative localization approaches. Even though UWB offers a decimeter level accuracy in line-of-sight (LoS) ranging, its accuracy degrades significantly in non-line-of-sight (NLoS) due to the significant unknown positive bias in the measurements. Thus, the measurement models for the UWB LoS and NLoS ranging conditions are different, and proper processing of NLoS measurements requires a bias compensation measure. The state-

of-art bias mitigation methods require a large amount of prior information and are hard to implement in real-time settings. Therefore, we aim to develop algorithmic bias compensation methods for UWB measurement under non-line-of-sight conditions such that they can be implemented in real-time on small portable computing boards. On the other hand, in practice, the measurement modal discriminators determine the type of UWB range measurements with only some level of certainty. To take into account the probabilistic nature of the NLoS identifiers, we aim to employs an interacting multiple model (IMM) estimator with in proposed cooperative localization framework.

The last part of this dissertation work design of proper wireless communication protocol for UWB transceivers such that they can be used as infrastructure-free communication medium in cooperative localization. The default communication protocols of the off-the-shelf UWB devices in the market are half-duplex, meaning that this transceivers *cannot* transmit (TX mode) and receive (RX mode) data packets at the same time. In other words, packets transmitted to a device in transmitter mode will be lost even there is no interference during the propagation if the intended receiver is in transmitting mode. Our objective in this work is to design a dynamic TDMA scheme as medium access control (MAC) protocol for ultra-wideband (UWB) communication. This dynamic TDMA protocol divides the channel access into different time slots and dynamically changes the time schedule such that the time schedule of the agents accommodates the change of the network topology because of the agents' mobility. To improve energy efficiency of our inter-agent communication, our objective is to use a negotiation-based rescheduling method motivated by the sensor protocols information via negotiation (SPIN) protocol to schedule cooperative localization updates selectively to reduce the communication cost while maintaining an acceptable level of localization performance. We further improve our communication and sensing resource management by developing a deep neural network (DNN) based measurement scheduling method.

1.4 Notations

We denote the vectors with bold small font. The p^{th} element of vector $\mathbf{x}$ is denoted by x_p or $[\mathbf{x}]_p$. We denote the inner product of two vectors $\mathbf{x}$ and $\mathbf{y}$ with appropriate dimensions by $\mathbf{x}.\mathbf{y}$. Furthermore, we use $\mathbf{1}$ as a vector of ones with appropriate dimension. We denote sets with capital curly font. For a $\mathcal{P} = \{1, \cdots, n\}$ and a vector $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$ with $0 \leq x_p \leq 1$, the set $\mathcal{R}_{\mathbf{x}}$ is a random set where $p \in \mathcal{P}$ is in $\mathcal{R}_{\mathbf{x}}$ with the probability x_p. Hence, we call such vector $\mathbf{x}$ as membership probability vector. Furthermore, for $\mathcal{S} \subset \mathcal{P}$, $\mathbf{1}_{\mathcal{S}} \in \{0,1\}^n$ is the vector whose p^{th} element is 1 if $p \in \mathcal{S}$ and 0 otherwise; we call $\mathbf{1}_{\mathcal{S}}$ the membership indicator vector of set $\mathcal{S}$. $|x|$ is the absolute value of $x \in \mathbb{R}$. By overloading the notation, we also use $|\mathcal{S}|$ as the cordiality of set $\mathcal{S}$. We denote a graph by $\mathcal{G}(\mathcal{A}, \mathcal{E})$ where $\mathcal{A}$ is the node set and $\mathcal{E} \subset \mathcal{A} \times \mathcal{A}$ is the edge set. $\mathcal{G}$ is undirected if and only $(i,j) \in \mathcal{E}$ means that agents i and j can exchange information. An undirected graph is connected if there is a path from each node to every other node in the graph. We denote the set of the neighboring nodes of node i by $\mathcal{N}_i = \{j \in \mathcal{A} | (i,j) \in \mathcal{E}\}$. We also use $d(\mathcal{G})$ to show the diameter of the graph. Given a set $\mathcal{F} \subset \mathcal{X} \times \mathbb{R}$ and an element $(p, \alpha) \in \mathcal{X} \times \mathbb{R}$ we define the addition operator $\oplus$ as $\mathcal{F} \oplus \{(p,\alpha)\} = \{(u,\gamma) \in \mathcal{X} \times \mathbb{R} \,|\, (u,\gamma) \in \mathcal{F}, u \neq p\} \cup \{(u, \gamma + \alpha) \in \mathcal{X} \times \mathbb{R} \,|\, (u,\gamma) \in \mathcal{F}, u = p\} \cup \{(p,\alpha) \in \mathcal{X} \times \mathbb{R} \,|\, (p,\gamma) \notin \mathcal{F}\}$. Given a collection of sets $\mathcal{F}_j \in \mathcal{X} \times \mathbb{R}$, $j \in \mathcal{N}$, we define the max-operation over these collection as $\underset{j \in \mathcal{N}}{\text{MAX}}\, \mathcal{F}_j = \{(u,\gamma) \in X \times \mathbb{R} | (u,\gamma) \in \bar{\mathcal{F}} \text{ s.t. } \gamma = \max_{(u,\alpha) \in \bar{\mathcal{F}}} \alpha\}$, where $\bar{\mathcal{F}} = \bigcup_{j \in \mathcal{N}} \mathcal{F}_j$.

We denote the standard Euclidean norm of vector $\mathbf{x} \in \mathbb{R}^n$ by $\|\mathbf{x}\| = \sqrt{\mathbf{x}^\top \mathbf{x}}$, and the (essential) supremum norm of a signal $f : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ by $\|f\|_{\text{ess}} = (\text{ess})\sup\{\|f(t)\|, t \geq 0\}$. The set of measurable essentially bounded functions $f : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is denoted by $\mathcal{L}^\infty_n$. The set of measurable functions $f : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ that satisfy $\int_0^t \|f(\tau)\| \mathrm{d}\tau < \infty$ is denoted by $\mathcal{L}^1_n$. For sets $\mathcal{A}$ and $\mathcal{B}$, the relative complement of $\mathcal{B}$ in $\mathcal{A}$ is $\mathcal{A} \backslash \mathcal{B} = \{x \in \mathcal{A} \,|\, x \notin \mathcal{B}\}$. For a vector $\mathbf{x} \in \mathbb{R}^n$, the sum of its elements is $\texttt{sum}(\mathbf{x})$. In a network of N agents, to emphasize that a variable is local to an agent $i \in\in \{1, \cdots, N\}$, we use superscripts. Moreover, if $p^i \in \mathbb{R}$ is a variable of agent $i \in\in \{1, \cdots, N\}$, the aggregated p^i's of the network is the

vector $\mathbf{p} = [\{p^i\}_{i=1}^N] = [p^1, \cdots, p^N]^\top \in \mathbb{R}^N$.

We denote $2^{\mathcal{A}}$ to be the set of all the subsets of a set $\mathcal{A}$. For a set $\mathcal{B}$ and a singleton set $\{a\}$, for simplicity we represent $\mathcal{B} \cup \{a\}$ by $\mathcal{B} \cup a$. Given an event set $\mathcal{V}$ and $e \in \mathcal{V}$, $P(e) : \mathcal{V} \to [0, 1]$ denotes the probability of event e happening. We denote a sequence of m increasing real numbers $(t_1, \cdots, t_m)$ (i.e., $t_k \leq t_{k+1}$ for $k \in \{1, \cdots, m\}$) by $(t)_1^m$. Given $(t)_1^n$ and $(v)_1^m$ we denote by $(t)_1^n \oplus (v)_1^m$ their concatenated increasing sequence, i.e., for $(u)_1^{n+m} = (t)_1^n \oplus (v)_1^m$ we have that any u_k, $k \in \{1, \cdots, n+m\}$ is either in $(t)_1^n$ or $(v)_1^m$ or is in both of $(t)_1^n$ and $(v)_1^m$. We assume that $(u)_1^{n+m}$ preserves the relative labeling of $(t)_1^n$ or $(v)_1^m$, i.e., if t_k and t_{k+1}, $k \in \{1, \cdots, n-1\}$ (resp. v_k and v_{k+1}, $k \in \{1, \cdots, m-1\}$) correspond to u_i and u_j in $(u)_1^{n+m}$, then $i < j$.

1.5 Preliminaries

1.5.1 Graph Theory

We review some basic concepts from algebraic graph theory following [120]. A weighted directed graph (digraph) is a triplet $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V} = \{1, \ldots, N\}$ is the *node set*, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the *edge set* and $\mathbf{A} = [\mathsf{a}_{ij}] \in \mathbb{R}^{N \times N}$ is a weighted *adjacency* matrix with the property that $\mathsf{a}_{ij} > 0$ if $(i, j) \in \mathcal{E}$ and $\mathsf{a}_{ij} = 0$, otherwise. A weighted digraph is *undirected* if $\mathsf{a}_{ij} = \mathsf{a}_{ji}$ for all $i, j \in \mathcal{V}$. An edge from i to j, denoted by (i, j), means that agent j can send information to agent i. For an edge $(i, j) \in \mathcal{E}$, i is called an *in-neighbor* of j and j is called an *out-neighbor* of i. We denote the set of the out-neighbors of an agent $i \in \mathcal{V}$ by $\mathcal{N}_{\text{out}}^i$. We define $\mathcal{N}_{\text{out}+i}^i = \mathcal{N}_{\text{out}}^i \cup \{i\}$. A *directed path* is a sequence of nodes connected by edges. A digraph is called *strongly connected* if for every pair of vertices there is a directed path connecting them. We refer to a strongly connected and undirected graph as a *connected graph*. The *weighted out-degree* and *weighted in-degree* of a node i, are

respectively, $\mathsf{d}_{\text{in}}^i = \sum_{j=1}^N \mathsf{a}_{ji}$ and $\mathsf{d}_{\text{out}}^i = \sum_{j=1}^N \mathsf{a}_{ij}$. A digraph is *weight-balanced* if at each node $i \in \mathcal{V}$, the weighted out-degree and weighted in-degree coincide (although they might be different across different nodes). The *(out-) Laplacian* matrix is $\mathsf{L} = [\ell_{ij}]$ is $\mathsf{L} = \mathsf{D}^{\text{out}} - \mathsf{A}$, where $\mathsf{D}^{\text{out}} = \text{Diag}(\mathsf{d}_{\text{out}}^1, \cdots, \mathsf{d}_{\text{out}}^N) \in \mathbb{R}^{N\times N}$. Note that $\mathsf{L}\mathbf{1}_N = \mathbf{0}$. A digraph is weight-balanced if and only if $\mathbf{1}_N^\top \mathsf{L} = \mathbf{0}$. For a strongly connected and weight-balanced digraph, $\text{rank}(\mathsf{L}) = N-1$, $\text{rank}(\mathsf{L}+\mathsf{L}^\top) = N-1$, and L has one zero eigenvalue $\lambda_1 = 0$ and the rest of its eigenvalues have positive real parts. We let $\mathsf{R} \in \mathbb{R}^{N\times(N-1)}$ be a matrix whose columns are normalized orthogonal complement of $\mathbf{1}_N$. Then

$$\mathbf{T}^\top \mathsf{L} \mathbf{T} = \begin{bmatrix} 0 & \mathbf{0} \\ 0 & \mathsf{L}^+ \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1}_N & \mathsf{R} \end{bmatrix}, \quad \mathsf{L}^+ = \mathsf{R}^\top \mathsf{L} \mathsf{R}. \tag{1.2}$$

For a strongly connected and weight-balanced digraph, $-\mathsf{L}^+$ is a Hurwitz matrix.

1.5.2 Concave Functions and Series

We develop the general auxiliary results below to use in the proof of Theorem of chapter 2. These results show some of the properties of the sum of evaluation of a concave and increasing function over increasing sequences and their concatenation. The decreasing sequence $(\delta\mathfrak{t})_1^n$ *majorizes* the decreasing sequence $(\delta\mathfrak{v})_1^n$, if

$$\delta\mathfrak{t}_1 \geq \delta\mathfrak{t}_2 \geq \cdots \geq \delta\mathfrak{t}_n,$$

$$\delta\mathfrak{v}_1 \geq \delta\mathfrak{v}_2 \geq \cdots \geq \delta\mathfrak{v}_n,$$

$$\delta\mathfrak{t}_1 + \cdots + \delta\mathfrak{t}_i \geq \delta\mathfrak{v}_1 + \cdots + \delta\mathfrak{v}_i, \ \ \forall i \in \{1, \cdots, n-1\}$$

and

$$\delta\mathfrak{t}_1 + \cdots + \delta\mathfrak{t}_n = \delta\mathfrak{v}_1 + \cdots + \delta\mathfrak{v}_n$$

hold.

Theorem 1.5.1 (Karamata's inequality [27]). *Let $f : \mathbb{R} \to \mathbb{R}$ be a concave and increasing function. Then, if sequence $(\delta t)_1^n$ majorizes $(\delta v)_1^n$, then $f(\delta t_1)+\cdots+f(\delta t_n) \leq f(\delta v_1)+\cdots+f(\delta v_n)$ holds.*

Lemma 1.5.1. *Let $f : \mathbb{R} \to \mathbb{R}$ be a concave and increasing function with $f(0) = 0$. If sequences $(\delta \mathfrak{t})_1^n$ and $(\delta \mathfrak{v})_1^m$ with $n \leq m$ satisfy $\delta \mathfrak{t}_1 + \cdots + \delta \mathfrak{t}_i \geq \delta \mathfrak{v}_1 + \cdots + \delta \mathfrak{v}_i, \quad \forall i \in \{1, \cdots, n-1\}$ and $\delta \mathfrak{t}_1 + \cdots + \delta \mathfrak{t}_n = \delta \mathfrak{v}_1 + \cdots + \delta \mathfrak{v}_m$ then*

$$f(\delta \mathfrak{t}_1) + \cdots + f(\delta \mathfrak{t}_n) \leq f(\delta \mathfrak{v}_1) + \cdots + f(\delta \mathfrak{v}_m)$$

holds.

Proof. We note that the sequence $(\delta \mathfrak{u})_1^m$ defined as $\delta \mathfrak{u}_i = \delta \mathfrak{u}_i$ for $i \in \{1, \cdots, n\}$ and $\delta \mathfrak{u}_i = 0$ for $i \in \{n+1, \cdots, m\}$ majorizes any sequence $(\delta \mathfrak{v})_1^m$ defined in the lemma statement. Then, since $f(0) = 0$, the proof follows from the Karamata's inequality [27]. □

Corollary 1.5.1. *Let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a monotone increasing and concave function. Then for any $a, b, c, d \in \mathbb{R}_{\geq 0}$ such that $0 \leq a \leq c$ and $0 \leq b \leq d$, then*

$$f(c) + f(d) - f(c+d) \leq f(a) + f(b) - f(a+b)$$

holds.

Proof. The assumption is that $a \leq c$ and $b \leq d$. Therefor, we have $a + b \leq c + d$. By taking $a, b, c+d$ and $c, d, a+b$ as $\delta \mathfrak{t}$'s $\delta \mathfrak{v}$'s respectively. There will be two possible cases for $\delta \mathfrak{t}$'s as

$$(A1): \quad \delta \mathfrak{t}_1 = c + d, \;\; \delta \mathfrak{t}_2 = a, \;\; \delta \mathfrak{t}_3 = b,$$

$$(A2): \quad \delta \mathfrak{t}_1 = c + d, \;\; \delta \mathfrak{t}_2 = b, \;\; \delta \mathfrak{t}_3 = a,$$

and there will be six possible cases for $\delta\mathfrak{v}$'s as

$$(B1): \quad \delta\mathfrak{v}_1 = a+b, \ \ \delta\mathfrak{v}_2 = d, \ \ \delta\mathfrak{v}_3 = c,$$

$$(B2): \quad \delta\mathfrak{v}_1 = a+b, \ \ \delta\mathfrak{v}_2 = c, \ \ \delta\mathfrak{v}_3 = d,$$

$$(B3): \quad \delta\mathfrak{v}_1 = d, \ \ \delta\mathfrak{v}_2 = a+b, \ \ \delta\mathfrak{v}_3 = c,$$

$$(B4): \quad \delta\mathfrak{v}_1 = c, \ \ \delta\mathfrak{v}_2 = a+b, \ \ \delta\mathfrak{v}_3 = d,$$

$$(B5): \quad \delta\mathfrak{v}_1 = c, \ \ \delta\mathfrak{v}_2 = d, \ \ \delta\mathfrak{v}_3 = a+b,$$

$$(B6): \quad \delta\mathfrak{v}_1 = d, \ \ \delta\mathfrak{v}_2 = c, \ \ \delta\mathfrak{v}_3 = a+b.$$

Taking any cases of A or B, we have $\delta\mathfrak{t}_1 + \delta\mathfrak{t}_2 + \delta\mathfrak{t}_3 = \delta\mathfrak{v}_1 + \delta\mathfrak{v}_2 + \delta\mathfrak{v}_3 = a+b+c+d$. Comparing any cases of A with any cases of B, $\delta\mathfrak{t}_1 \geq \delta\mathfrak{v}_1$. Taking case $(A1)$, since $a > b$ then we have $c+d+a \geq a+b+d$ and $c+d+a \geq a+b+c$ and also simply we have $c+d+a \geq c+d$. Therefor, Taking case $A1$ and comparing with any cases of B, we have $\delta\mathfrak{t}_1 + \delta\mathfrak{t}_2 \geq \delta\mathfrak{v}_1 + \delta\mathfrak{v}_2$. The same reasoning also can be done for case $A2$. Hence taking any cases of A and B, we know that $\delta\mathfrak{t}_1, \delta\mathfrak{t}_2, \delta\mathfrak{t}_3$ majorizes $\delta\mathfrak{v}_1, \delta\mathfrak{v}_2, \delta\mathfrak{v}_3$. This results in

$$f(c) + f(d) + f(a+b) \leq f(a) + f(b) + f(c+d)$$

and consequently

$$f(a) + f(b) - f(a+b) \leq f(c) + f(d) - f(c+d).$$

□

Lemma 1.5.2. *For any* $(\mathfrak{q})_1^l$, *let*

$$g((\mathfrak{q})_1^l) = \sum_{i=1}^{l-1} f(\Delta\mathfrak{q}_i),$$

where $\Delta\mathfrak{q}_i = \mathfrak{q}_{i+1} - \mathfrak{q}_i$ *and* f *be a concave and increasing function with* $f(0) = 0$. *Now, consider two increasing sequences* $(\mathfrak{t})_1^n$ *and* $(\mathfrak{u})_1^l$, *and their concatenation* $(\mathfrak{a})_1^{n+l} = (\mathfrak{t})_1^n \oplus (\mathfrak{u})_1^l$. *Then,*

$$g((\mathfrak{a})_1^{n+l}) - g((\mathfrak{t})_1^n) \geq 0.$$

holds

Proof. If $\mathfrak{a}_p = t_1$ and $\mathfrak{a}_q = t_n$, then since $(\mathfrak{a})_1^{n+l}$ is a increasing sequence, $p < q$. Let the subsequence of $(\mathfrak{a})_1^{n+l}$ ranging from index p to q be $(\mathfrak{v})_1^m$ where $m \geq n$. Letting $\Delta\mathfrak{v}_i = \mathfrak{v}_{i+1} - \mathfrak{v}_i$ and $\Delta\mathfrak{t}_i = \mathfrak{t}_{i+1} - \mathfrak{t}_i$, we rearrange $\Delta\mathfrak{v}_i$'s and $\Delta\mathfrak{t}_i$'s in a descending order to form the sequences $(\delta\mathfrak{v})_1^{l-1}$ and $(\delta\mathfrak{t})_1^{n-1}$. Since $\mathfrak{a}_p = \mathfrak{t}_1$ and $\mathfrak{a}_q = \mathfrak{t}_n$, we have

$$\sum_{i=1}^{m-1} \Delta\mathfrak{v}_i = \sum_{i=1}^{m-1} \delta\mathfrak{v}_i = \sum_{i=1}^{n-1} \delta\mathfrak{t}_i = \sum_{i=1}^{n-1} \Delta\mathfrak{t}_i = \mathfrak{t}_n - \mathfrak{t}_1.$$

Because $(\mathfrak{a})_1^{n+l} = (\mathfrak{t})_1^n \oplus (\mathfrak{u})_1^l$, then $\forall i \in \{1, \cdots, n\}$ there exists $\mathsf{S}_i \subset \{1, \cdots, m\}$ such that $\sum_{j \in \mathsf{S}_i} \delta\mathfrak{v}_j = \delta\mathfrak{t}_i$, where $\mathsf{S}_i \cap \mathsf{S}_k = \emptyset$, $i \neq k$. Consequently, for $r \in \{1, \cdots, m\}$, we have $\sum_{i=1}^r \delta\mathfrak{v}_i = \sum_{j \in \mathsf{S}} \delta\mathfrak{t}_j$ for $\mathsf{S} \subset \{1, \cdots, n\}$ and $|S| \leq r$. Since $(\delta\mathfrak{t})_1^{n-1}$ is a decreasing sequence, we can write

$$\sum\nolimits_{i=1}^r \delta\mathfrak{v}_i \leq \sum\nolimits_{i=1}^r \delta\mathfrak{t}_i.$$

Thus,

$$f(\delta\mathfrak{t}_1) + \cdots + f(\delta\mathfrak{t}_{n-1}) \leq f(\delta\mathfrak{v}_1) + \cdots + f(\delta\mathfrak{v}_{m-1})$$

holds as a result of Lemma 1.5.1. Given that

$$f(\delta\mathfrak{t}_1) + \cdots + f(\delta\mathfrak{t}_{n-1}) = \sum_{i=1}^{n-1} f(\Delta\mathfrak{t}_i)$$

and

$$f(\delta\mathfrak{v}_1) + \cdots + f(\delta\mathfrak{v}_{m-1}) = \sum_{i=1}^{m-1} f(\Delta\mathfrak{v}_i) \leq \sum_{i=1}^{n+1-1} f(\Delta\mathfrak{a}_i),$$

then

$$\sum_{i=1}^{n-1} f(\Delta \mathfrak{t}_i) \leq \sum_{i=1}^{n+1-1} f(\Delta \mathfrak{a}_i),$$

which concludes the proof. □

Lemma 1.5.3. *For any* $(\mathfrak{q})_1^l$*, let*

$$g((\mathfrak{q})_1^l) = \sum\nolimits_{i=1}^{l} f(\Delta \mathfrak{q}_i)$$

where $\Delta \mathfrak{q}_i = \mathfrak{q}_{i+1} - \mathfrak{q}_i$ *and* f *is a concave and increasing function with* $f(0) = 0$*. Now, consider three increasing sequences* $(\mathfrak{t})_1^n$ *and* $(\mathfrak{v})_1^m$ *and* $(\mathfrak{u})_1^l$ *and concatenations* $(\mathfrak{a})_1^{n+l} = (\mathfrak{t})_1^n \oplus (\mathfrak{u})_1^l$ *and* $(\mathfrak{b})_1^{m+l} = (\mathfrak{v})_1^m \oplus (\mathfrak{u})_1^l$ *where* $(\mathfrak{v})_1^m$ *is a sub-sequence of* $(\mathfrak{t})_1^n$*, then*

$$\big(g((\mathfrak{b})_1^{m+l}) - g((\mathfrak{v})_1^m)\big) - \big(g((\mathfrak{a})_1^{n+l}) - g((\mathfrak{t})_1^n)\big) \geq 0.$$

Proof. Let the sequence $(\mathfrak{u})_1^p$ be the first p elements of $(\mathfrak{u})_1^l$. Then, we can form

$$\begin{aligned} \Delta S_p = &\big(g((\mathfrak{v})_1^m \oplus (\mathfrak{u})_1^p) - g((\mathfrak{v})_1^m \oplus (\mathfrak{u})_1^{p-1})\big) \\ &- \\ &\big(g((\mathfrak{t})_1^n \oplus (\mathfrak{u})_1^p) - g((\mathfrak{t})_1^n \oplus (\mathfrak{u})_1^{p-1})\big), \end{aligned}$$

where $(\mathfrak{u})_1^0$ to be an empty sequence with no members. Since $(\mathfrak{v})_1^m$ is a sub-sequence of $(\mathfrak{t})_1^n$ and $(\mathfrak{u})_1^p$ having one member more over $(\mathfrak{u})_1^{p-1}$, then we have

$$\begin{aligned} \Delta S_p = &(f(\Delta S_1) + f(\Delta S_2) - f(\Delta S_1 + \Delta S_2)) \\ &- \\ &(f(\Delta S_3) + f(\Delta S_4) - f(\Delta S_3 + \Delta S_4)) \end{aligned}$$

with $0 \leq \Delta S_3 \leq \Delta S_1$ and $0 \leq \Delta S_4 \leq \Delta S_2$. From Corollary 1.5.1, we can conclude that

$\Delta S_p \geq 0$. Then, given

$$\sum_{p=1}^{l} \Delta S_p = \big(g((\mathfrak{b})_1^{m+l}) - g((\mathfrak{v})_1^m)\big) - \big(g((\mathfrak{a})_1^{n+l}) - g((\mathfrak{v})_1^m)\big),$$

the proof is concluded. □

1.5.3 Submodular Functions

A set function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ is monotone increasing if $f(\mathcal{P}_1) \leq f(\mathcal{P}_2)$, and submodular if for any $p \in \mathcal{P} \setminus \mathcal{P}_2$,

$$\Delta_f(p|\mathcal{P}_1) \geq \Delta_f(p|\mathcal{P}_2), \tag{1.3}$$

hold for any sets $\mathcal{P}_1 \subset \mathcal{P}_2 \subset \mathcal{P}$. The total curvature of a submodular set function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, which shows the worst-case increase in the value of the function when member p is added, is defined as

$$c = 1 - \min_{\mathcal{S}, p \notin \mathcal{S}} \frac{\Delta_f(p|\mathcal{S})}{\Delta_f(p|\emptyset)} \tag{1.4}$$

Note that $c \in [0, 1]$; the curvature of $c = 0$ means that the function is modular, i.e., $f(\{p_1, p_2\}) = f(\{p_1\}) + f(\{p_2\})$, $p_1, p_2 \in \mathcal{P}$, while $c = 1$ means that there is at least a member that adds no value to function f in a special circumstance. Curvature c represents a measure of the diminishing return of a set function. Whenever the total curvature is not known, it is rational to assume the worst case scenario and set $c = 1$.

In the rest of this work, without loss of generality, we assume that the ground set $\mathcal{P}$ is equal to $\{1, \cdots, n\}$. For a submodular function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, its *multilinear extension*

$F : [0,1]^n \to \mathbb{R}_{\geq 0}$ in the continuous space is

$$F(\mathbf{x}) = \sum_{\mathcal{R} \subset \mathcal{P}} f(\mathcal{R}) \prod_{p \in \mathcal{R}} [\mathbf{x}]_p \prod_{p \notin \mathcal{R}} (1 - [\mathbf{x}]_p), \quad \mathbf{x} \in [0,1]^n. \tag{1.5}$$

F in (1.5) is indeed equivalent to

$$F(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_\mathbf{x})], \tag{1.6}$$

where $\mathcal{R}_\mathbf{x} \subset \mathcal{P}$ is a set where each element $p \in \mathcal{R}_\mathbf{x}$ appears independently with the probabilities $[\mathbf{x}]_p$. Then, taking the derivatives of $F(\mathbf{x})$ yields

$$\frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_\mathbf{x} \cup \{p\}) - f(\mathcal{R}_\mathbf{x} \setminus \{p\})], \tag{1.7}$$

and

$$\begin{aligned} \frac{\partial^2 F}{\partial [\mathbf{x}]_p \partial [\mathbf{x}]_q}(\mathbf{x}) = \mathbb{E}[&f(\mathcal{R}_\mathbf{x} \cup \{p,q\}) - f(\mathcal{R}_\mathbf{x} \cup \{q\} \setminus \{p\}) \\ &- f(\mathcal{R}_\mathbf{x} \cup \{p\} \setminus \{q\}) + f(\mathcal{R}_\mathbf{x} \setminus \{p,q\})]. \end{aligned} \tag{1.8}$$

Given a ground set $\mathcal{P}$, a matroid is defined as the pair $\mathcal{M} = \{\mathcal{P}, \mathcal{I}\}$ with $\mathcal{I} \subset 2^\mathcal{P}$ such that (a) for any $\mathcal{B} \in \mathcal{I}$ and $\mathcal{A} \subset \mathcal{B}$ then $\mathcal{A} \in \mathcal{I}$, and (b) for any $\mathcal{A}, \mathcal{B} \in \mathcal{I}$ and $|\mathcal{B}| > |\mathcal{A}|$, then there exists $x \in \mathcal{B} \setminus \mathcal{A}$ such that $\mathcal{A} \cup x \in \mathcal{I}$.

- $\mathcal{B} \in \mathcal{I}$ and $\mathcal{A} \subset \mathcal{B}$ then $\mathcal{A} \in \mathcal{I}$
- $\mathcal{A}, \mathcal{B} \in \mathcal{I}$ and $|\mathcal{B}| > |\mathcal{A}|$, then there exists $x \in \mathcal{B} \setminus \mathcal{A}$ such that $\mathcal{A} \cup x \in \mathcal{I}$

Partition matroid $\mathcal{M}$ is defined as $\mathcal{M} = \{\mathcal{R} \subset \mathcal{P} \big| \ |\mathcal{R} \cap \mathcal{P}_i| \leq k_i, \forall i \in \mathcal{A}\}$, $1 \leq \kappa_i \leq |\mathcal{P}_i|$ and $\mathcal{A} = \{1, \ldots, N\}$, with $\bigcup_{i \in \mathcal{A}} \mathcal{P}_i = \mathcal{P}$ and $\mathcal{P}_i \cap \mathcal{P}_j = \emptyset$, $i \neq j$. The *matroid polytop* for

partition matroid is a convex hull defined as (with abuse of notation)

$$\mathcal{M} = \{\mathbf{x} \in [0,1]^n \mid \sum\nolimits_{p\in\mathcal{P}_i} [\mathbf{x}_i]_p \leq \kappa_i, \forall i \in \mathcal{A}\}. \tag{1.9}$$

where $[\mathbf{x}]_p$, $p \in \mathcal{P}_i$ associated with membership probability of strategies in $\mathcal{P}_i$, $i \in \mathcal{A} = \{1, \cdots, N\}$, we have $\sum_{p\in\mathcal{P}_i}[\mathbf{x}]_p \leq \kappa_i$.

In the following results we derive some auxiliary results on the first and the second order derivatives of the multilinear extension F.

Lemma 1.5.4 (Second derivative of multi linear extension for general case)**.** *Consider the set value optimization problem* (4.1)*. Suppose $f : \mathcal{P} \to \mathbb{R}_{\geq 0}$ is an increasing and submodular set function and consider its multilinear extension $F : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$. Then*

$$\left|\frac{\partial^2 F}{\partial x_p \partial x_q}\right| \leq f(\mathcal{S}^\star), \qquad p, q \in \mathcal{P}.$$

Proof. Since $p \notin \mathcal{R}_\mathbf{x} \cup \{q\} \setminus \{p\}$, therefor by submodularity of f we can write

$$\begin{aligned} &0 \leq \Delta_f(p|\mathcal{R}_\mathbf{x} \cup \{q\} \setminus \{p\}) \leq f(\{p\}), \\ &0 \leq \Delta_f(p|\mathcal{R}_\mathbf{x}\backslash\{p,q\}) \leq f(\{p\}). \end{aligned} \tag{1.10}$$

Knowing that

$$\Delta_f(p|\mathcal{R}_\mathbf{x}\cup\{q\}\setminus\{p\}) = f(\mathcal{R}_\mathbf{x} \cup \{p,q\}) - f(\mathcal{R}_\mathbf{x} \cup \{q\}\backslash\{p\}),$$

and

$$\Delta_f(p|\mathcal{R}_\mathbf{x}\backslash\{p,q\}) = f(\mathcal{R}_\mathbf{x} \cup \{p\})\backslash\{q\}) - f(\mathcal{R}_\mathbf{x}\backslash\{p,j\}),$$

then definition (4.5) can be written as

$$\frac{\partial F}{\partial x_p \partial x_q} = \mathbb{E}[\Delta_f(p|\mathcal{R}_{\mathbf{x}} \cup \{q\}\backslash\{p\}) - \Delta_f(p|\mathcal{R}_{\mathbf{x}}\backslash\{p,q\})]. \tag{1.11}$$

Putting (1.10) and (1.11) together results in

$$\left|\frac{\partial^2 F}{\partial x_p \partial x_q}\right| \leq f(\{p\}).$$

where knowing that $f(\{p\}) \leq f(\mathcal{S}^\star)$ concludes the proof. □

Lemma 1.5.5 (First and second derivatives of the multilinear extension for a known curvature)**.** *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, $\mathcal{P} = \{1, \cdots, n\}$, be increasing and submodular set function with curvature c, and the multinear extension function $F(\mathbf{x})$ defined in* (4.2)*. Then, $\frac{\partial F}{\partial [\mathbf{x}]_p} \geq (1-c)f(p)$ for all $p \in \mathcal{P}$ and $\mathbf{x} \in [0,1]^n$. Moreover, $-cf(\mathcal{R}^\star) \leq \frac{\partial^2 F}{\partial [\mathbf{x}]_p [\mathbf{x}]_q} \leq 0$ for all $p, q \in \mathcal{P}$ and $\mathbf{x} \in [0,1]^n$.*

Proof. The derivative of $F(x)$ can be written as

$$\frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}) = \Delta_f(p\,|\mathcal{R}_{\mathbf{x}} \setminus \{p\}). \tag{1.12}$$

Furthermore, by the definition of the total curvature (4.21) we can write $c \geq 1 - \frac{\Delta_f(p|\mathcal{R}_{\mathbf{x}}\backslash p)}{f(p)}$, and by conjunction with equation (1.12), we have $\frac{\partial F}{\partial [\mathbf{x}]_p} \geq (1-c)f(p)$ which proves the first part of Lemma. Since $p \notin \mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\}$, therefore by the definition of the total curvature (4.21) we can write

$$(1-c)f(\{p\}) \leq \Delta_f(p|\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p_i\}) \leq f(\{p\}). \tag{1.13}$$

Moreover, Since $p \notin \mathcal{R}_{\mathbf{x}} \setminus \{p,q\}$, therefore by the definition of the total curvature (4.21) we

can write

$$(1-c)f(\{p\}) \leq \Delta_f(p|\mathcal{R}_{\mathbf{x}} \setminus \{p,q\}) \leq f(\{p\}). \tag{1.14}$$

Knowing that $\Delta_f(p|\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\}) = f(\mathcal{R}_{\mathbf{x}} \cup \{p,q\}) - f(\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\})$ and $\Delta_f(p|\mathcal{R}_{\mathbf{x}} \setminus \{p,q\}) = f(\mathcal{R}_{\mathbf{x}} \cup \{p\}) \setminus \{q\}) - f(\mathcal{R}_{\mathbf{x}} \setminus \{p,q\})$, the definition of second order derivative of F (4.5), we can be written as

$$\frac{\partial F}{\partial [\mathbf{x}]_p \partial [\mathbf{x}]_q} = \mathbb{E}[\Delta_f(p|\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\}) - \Delta_f(p|\mathcal{R}_{\mathbf{x}} \setminus \{p,q\})]. \tag{1.15}$$

Putting (1.13) and (1.14) and (1.15) together in conjunction with submodular property of f results in $-cf(\{p\}) \leq \frac{\partial^2 F}{\partial [\mathbf{x}]_q \partial [\mathbf{x}]_q} \leq 0$. Knowing that $f(\{p\}) \leq f(\mathcal{R}^\star)$ results in proving the second part of Lemma. □

Lemma 1.5.6 (Directional Convexity)**.** *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, $\mathcal{P} = \{1, \cdots, n\}$, be monotone increasing and submodular set function with a multinear extension function $F(\mathbf{x})$ defined in (4.2). Then, for any given $\mathbf{x} \in [0,1]^n$ and $\mathbf{w} \in \{-1, 0, 1\}^n$ where $w_p = 1,\ w_q = -1$ and $w_l = 0,\ l \in \{1, \cdots, n\} \backslash \{p, q\}$ for some $p, q \in \{1, \cdots, n\}$, $F(\mathbf{x} + \lambda \mathbf{w}) : \mathbb{R} \to \mathbb{R}_{\geq 0}$ is a convex function of λ.*

Proof. Defining the vector $\mathbf{w} \in \mathbb{R}^n$ and $w_p = 1,\ w_q = -1$ and $w_l = 0,\ l \neq p, q$, then the multilinear extension of set function f in the direction of $\mathbf{w}$ is defined as

$$\begin{aligned}
F(\mathbf{x} + \lambda \mathbf{w}) = \\
&\sum_{\mathcal{R} \subset \mathcal{P} \setminus \{p,q\}} f(\{p\} \cup \mathcal{R})([\mathbf{x}]_p + \lambda)(1 - ([\mathbf{x}]_q - \lambda))\mathbb{P}(\mathcal{R}) + \\
&\qquad f(\{q\} \cup \mathcal{R})(1 - ([\mathbf{x}]_p + \lambda))([\mathbf{x}]_q - \lambda)\mathbb{P}(\mathcal{R}) + \\
&\qquad f(\mathcal{R})(1 - ([\mathbf{x}]_p + \lambda))(1 - ([\mathbf{x}]_q - \lambda))\mathbb{P}(\mathcal{R}) + \\
&\qquad f(\mathcal{R}) - f(\{p,q\} \cup \mathcal{R}))([\mathbf{x}]_p + \lambda)([\mathbf{x}]_q - \lambda)\mathbb{P}(\mathcal{R}).
\end{aligned}$$

with $\mathbb{P}(\mathcal{R}) = \prod_{r\in\mathcal{R}} x_r \prod_{r\notin\mathcal{R}}(1-x_r)$. Taking the second derivative of $F(\mathbf{x}+\lambda\mathbf{w})$ with respect to λ yields

$$\frac{\partial^2 F(\mathbf{x}+\lambda\mathbf{w})}{\partial\lambda^2} = \sum_{\mathcal{R}\subset\mathcal{P}\setminus\{p,q\}} 2\mathbb{P}(\mathcal{R})(f(\{p\}\cup\mathcal{R}) + f(\{q\}\cup\mathcal{R}) - f(\mathcal{R}) - f(\{p,q\}\cup\mathcal{R})).$$

The submodularity of f asserts that $\frac{\partial^2 F(\mathbf{x}+\lambda\mathbf{w})}{\partial\lambda^2} \geq 0$ and consequently, $F(\mathbf{x}+\lambda\mathbf{w})$ is a convex function of λ. □

Lemma 1.5.7 (Interval Bound of Twice differentiable function)**.** *Consider a twice differentiable function $F(\mathbf{x}) : [0,1]^n \to \mathbb{R}$ which satisfies $\left|\frac{\partial^2 F}{\partial[\mathbf{x}]_p \partial[\mathbf{x}]_q}\right| \leq \alpha$ for any $p, q \in \{1, \cdots, n\}$. Then for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ satisfying $\mathbf{x}_2 \geq \mathbf{x}_1$ and $\mathbf{1}.(\mathbf{x}_2 - \mathbf{x}_1) \leq \beta$ we have*

$$\left|\frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_1 + \epsilon(\mathbf{x}_2 - \mathbf{x}_1)) - \frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_1)\right| \leq \epsilon\alpha\beta, \tag{1.16a}$$

$$F(\mathbf{x}_2) - F(\mathbf{x}_1) \geq \nabla F(\mathbf{x}_1).(\mathbf{x}_2 - \mathbf{x}_1) - \frac{1}{2}\alpha\beta^2, \tag{1.16b}$$

for $\epsilon \in [0\ 1]$.

Proof. Let $\mathbf{h}_p = [\frac{\partial^2 F}{\partial[\mathbf{x}]_p \partial x_1}, \cdots, \frac{\partial^2 F}{\partial[\mathbf{x}]_p \partial x_n}]^\top$. Then, we can write

$$\begin{aligned}
&\left|\frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_1 + \epsilon(\mathbf{x}_2 - \mathbf{x}_1)) - \frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_1)\right| \\
&= \left|\int_0^\epsilon \mathbf{h}_p(\mathbf{x}_1 + \tau(\mathbf{x}_2 - \mathbf{x}_1)).(\mathbf{x}_2 - \mathbf{x}_1)\mathrm{d}\tau\right| \\
&\leq \int_0^\epsilon \alpha\mathbf{1}.(\mathbf{x}_2 - \mathbf{x}_1)\mathrm{d}\tau = \epsilon\alpha\beta,
\end{aligned} \tag{1.17}$$

Furthermore, $F(\mathbf{x}_2) - F(\mathbf{x}_1) = \int_0^1 \nabla F(\mathbf{x}_1 + \epsilon(\mathbf{x}_2 - \mathbf{x}_1)).(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t))\mathrm{d}\epsilon \geq \int_0^1 (\nabla F(\mathbf{x}_1) - \epsilon\alpha\beta).(\mathbf{x}_2 - \mathbf{x}_1)\mathrm{d}\epsilon = \nabla F(\mathbf{x}_1).(\mathbf{x}_2 - \mathbf{x}_1) - \frac{1}{2}\alpha\beta^2$, with the third line follow from equation (1.17), which concludes the proof. □

Lemma 1.5.8. *Suppose $f : \mathcal{P} \to \mathbb{R}_{\geq 0}$ is an increasing and submodular set function and consider $\mathbf{x}$ to be a membership probability vector of set $\mathcal{Q} \subset \mathcal{P}$ with $\mathbf{1}.\mathbf{x} = 1$. We define $\mathcal{R}_{\mathbf{x}}$ to be the set resulted by sampling independently each member of $\mathcal{Q}$ according to probability vector $\mathbf{x}$ and $\mathcal{T}_{\mathbf{x}} = \{t\},\ t \in \mathcal{Q}$ to be a single member set which is chosen according to $\mathbf{x}$. the following holds for any random set $\mathcal{S} \in \mathcal{P} \setminus \mathcal{Q}$.*

$$\mathbb{E}[f(\mathcal{R}_{\mathbf{x}_{ii}} \cup S)] \leq \mathbb{E}[f(\mathcal{T}_{\mathbf{x}_{ii}} \cup S)].$$

Proof. Defining $\mathcal{R}_{\mathbf{x}} = \{r_1, \cdots, r_o\}$, then we can write

$$\begin{aligned}
&\mathbb{E}[f(\mathcal{R}_{\mathbf{x}} \cup \mathcal{S})] = \mathbb{E}[f(\mathcal{S}) + \sum_{l=1}^{o} \Delta_f(r_l|\mathcal{S} \cup \{r_1, \cdots, r_{l-1}\})] \\
&\leq \mathbb{E}[f(\mathcal{S}) + \sum_{r_l \in \mathcal{R}_{\mathbf{x}}} \Delta_f(r_l|\mathcal{S})] = \mathbb{E}_{\mathcal{S}}[\mathbb{E}_{\mathcal{R}_{\mathbf{x}}|\mathcal{S}}[f(\mathcal{S}) + \sum_{r_l \in \mathcal{R}_{\mathbf{x}}} \Delta_f(r_l|\mathcal{S})]] \\
&= \mathbb{E}_{\mathcal{S}}[f(\mathcal{S}) + \mathbb{E}_{\mathcal{R}_{\mathbf{x}}|\mathcal{S}}[\sum_{r_l \in \mathcal{R}_{\mathbf{x}}} \Delta_f(r_l|\mathcal{S})]] = \mathbb{E}_{\mathcal{S}}[f(\mathcal{S}) + \sum_{r_l \in \mathcal{Q}} x_l \Delta_f(r_l|\mathcal{S})] \\
&= \mathbb{E}_{\mathcal{S}}[f(\mathcal{S}) + \mathbb{E}_{\mathcal{T}_{\mathbf{x}}|\mathcal{S}}[\Delta_f(t|\mathcal{S})]] = \mathbb{E}_{\mathcal{S}}[\mathbb{E}_{\mathcal{T}_{\mathbf{x}}|\mathcal{S}}[f(\mathcal{S}) + \Delta_f(t|\mathcal{S})]] = \mathbb{E}[f(\mathcal{T}_{\mathbf{x}} \cup S)]
\end{aligned}$$

which concludes the proof. □

1.5.4 Stochastic Estimation of The Relaxed Functions' gradient

The stochastic interpretation (1.6) of the multilinear-extension and its derivatives leads to empirical estimation of $\nabla F(\mathbf{x}(t))$. Chernoff-Hoeffding's inequality can be used to quantify the quality of these estimates given the number of samples.

Theorem 1.5.2 (Chernoff-Hoeffding's inequality [121])**.** *Consider a set of K independent random variables $X_1, ..., X_K$ where $a < X_i < b$. Let $S_K = \sum_{i=1}^{K} X_i$, then*

$$\mathbb{P}[|S_K - \mathbb{E}[S_K]| > \delta] < 2\,\mathrm{e}^{-\frac{2\delta^2}{K(b-a)^2}},$$

for any $\delta \in \mathbb{R}_{\geq 0}$.

The following lemma, whose proof relies on the Chernoff-Hoeffding's inequality, can quantify the quality of estimating the gradient of a multilinear extension function by sampling from the ground set.

Lemma 1.5.9. *Suppose* $f : \mathcal{P} \to \mathbb{R}_{\geq 0}$ *is an increasing and submodular set function and consider its multilinear extension* $F : [0,1]^n \to \mathbb{R}_{\geq 0}$. *Let* $\widetilde{\nabla F}(\mathbf{x})$ *be the estimate of* $\nabla F(\mathbf{x})$ *that is calculated by taking* $K \in \mathbb{Z}_{>0}$ *samples of set* $\mathcal{R}_{\mathbf{x}}$ *according to membership probability vector* $\mathbf{x}$. *Then,*

$$\left| \left[\widetilde{\nabla F}(\mathbf{x})\right]_p - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}) \right| \geq \frac{1}{2T} f(\mathcal{R}^\star), \quad p \in \{1, \cdots, n\}, \tag{1.18}$$

with the probability of at least $2\mathrm{e}^{-\frac{1}{8T^2}K}$, *for any* $T \in \mathbb{Z}_{>0}$.

Proof. Define the random variable

$$X = \left((f(\mathcal{R}_{\mathbf{x}} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}} \backslash \{p\})) - \frac{\partial F}{\partial x_p}(\mathbf{x}) \right) \Big/ f(\mathcal{R}^\star),$$

and assume that we take K samples from $\mathcal{R}_{\mathbf{x}}$ to construct $\{X_k\}_{k=1}^K$ realization of X. Since f is a submodular function, then we have $(f(\mathcal{R}_{\mathbf{x}} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}} \backslash \{p\})) \leq f(\mathcal{R}^\star)$. Consequently using equation (1.7), we conclude that $0 \leq X \leq 1$. Hence, using Theorem 1.5.2, we have $\left| \sum_{k=1}^K X_k \right| \geq \frac{1}{2T} K$ with the probability of at least $2\mathrm{e}^{-\frac{1}{8T^2}K}$. Hence, the estimation accuracy of $\nabla F(\mathbf{x})$, is given by $\left| \left[\widetilde{\nabla F}(\mathbf{x})\right]_p - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}) \right| \geq \frac{1}{2T} f(\mathcal{R}^\star),\ p \in \{1, \cdots, n\}$ with the probability of at least $2\mathrm{e}^{-\frac{1}{8T^2}K}$. □

1.5.5 Linear Systems Results

Lemma 1.5.10. *Let* $\mathbf{L}$ *be the Laplacian matrix of a strongly connected and weight-balanced digraph. Recall* $\mathbf{L}^+ = \mathbf{R}^\top \mathbf{L} \mathbf{R}$ *from* (1.2)*. Let* $\mathbf{g}(t) = [g_1(t), ..., g_n(t)]^\top \in \mathcal{L}_n^\infty$*. Then,*

$$\lim_{t\to\infty} \int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)} \mathbf{R}^\top \mathbf{L}\, \mathbf{g}(\tau) \mathrm{d}\tau = \mathbf{0}, \tag{1.19}$$

is guaranteed to hold if and only if

$$\lim_{t\to\infty} \int_0^t \mathrm{e}^{-(t-\tau)} g^i(\tau)\, \mathrm{d}\tau = \alpha \in \mathbb{R}, \quad i \in\in \{1, \cdots, N\}. \tag{1.20}$$

Proof. Let

$$\dot{\boldsymbol{\zeta}} = -\mathbf{L}^+ \boldsymbol{\zeta} + \mathbf{R}^\top \mathbf{L} \mathbf{g}(t), \qquad \boldsymbol{\zeta}(0) \in \mathbb{R}^{N-1}, \tag{1.21}$$

$$\dot{\boldsymbol{\eta}} = -\boldsymbol{\eta} + \mathbf{R}^\top \mathbf{L} \mathbf{g}(t), \qquad \boldsymbol{\eta}(0) \in \mathbb{R}^{N-1}. \tag{1.22}$$

The trajectories $t \mapsto \boldsymbol{\zeta}$ and $t \mapsto \boldsymbol{\eta}$ of these two dynamics for $t \in \mathbb{R}_{\geq 0}$ are given by

$$\boldsymbol{\zeta}(t) = \mathrm{e}^{-\mathbf{L}^+ t} \boldsymbol{\zeta}(0) + \int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)} \mathbf{R}^\top \mathbf{L} \mathbf{g}(\tau) \mathrm{d}\tau, \tag{1.23}$$

$$\boldsymbol{\eta}(t) = \mathrm{e}^{-t} \boldsymbol{\eta}(0) + \mathbf{R}^\top \mathbf{L} \int_0^t \mathrm{e}^{-(t-\tau)}\, \mathbf{g}(\tau) \mathrm{d}\tau. \tag{1.24}$$

Let $\mathbf{e} = \boldsymbol{\zeta} - \boldsymbol{\eta}$. Then, the error dynamics between (1.21) and (1.22) is given by

$$\dot{\mathbf{e}} = -\mathbf{e} + (\mathbf{I} - \mathbf{L}^+) \boldsymbol{\zeta}. \tag{1.25}$$

or equivalently

$$\dot{\mathbf{e}} = -\mathbf{L}^+ \mathbf{e} + (\mathbf{L}^+ + \mathbf{I}) \boldsymbol{\eta}. \tag{1.26}$$

Let (1.19) hold. Since $-\mathbf{L}^+$ is a Hurwitz matrix, we have $\lim_{t\to\infty}\boldsymbol{\zeta}(t)=0$. Moreover, since $\mathbf{g}$ is essentially bounded, the trajectories of $\boldsymbol{\zeta}$ are guaranteed to be bounded. Therefore, considering error dynamics (1.25), by invoking the ISS stability results [88], we have the guarantees that $\lim_{t\to\infty}\mathbf{e}(t)=\mathbf{0}$, and consequently $\lim_{t\to\infty}\boldsymbol{\eta}(t)=\mathbf{0}$. As such, from (1.24) we obtain

$$\mathbf{R}^\top\mathbf{L}\lim_{t\to\infty}\int_0^t \mathrm{e}^{-(t-\tau)}\mathbf{g}(\tau)\mathrm{d}\tau=\mathbf{0}. \tag{1.27}$$

The nullspace of $\mathbf{R}^\top\mathbf{L}\in\mathbb{R}^{(N-1)\times N}$ is spanned by $\mathbf{1}_N$, thus,

$$\lim_{t\to\infty}\int_0^t \mathrm{e}^{-(t-\tau)}\mathbf{g}(\tau)\mathrm{d}\tau=\alpha\mathbf{1}_N,\quad \alpha\in\mathbb{R},$$

which validates (1.20). Now let (1.20) hold. Then, using (1.24), we obtain $\lim_{t\to\infty}\boldsymbol{\eta}(t)=\mathbf{0}$. Since $\mathbf{g}$ is essentially bounded, the trajectories of $\boldsymbol{\zeta}$ are guaranteed to be bounded. Thereby, considering error dynamics (1.26), by invoking the ISS stability results [88], we have the guarantees that $\lim_{t\to\infty}\mathbf{e}(t)=\mathbf{0}$, and consequently $\lim_{t\to\infty}\boldsymbol{\eta}(t)=\mathbf{0}$. Since $-\mathbf{L}^+$ is a Hurwitz matrix, we obtain (1.19) from (1.23). □

Lemma 1.5.11. *Let $\mathbf{u}:\mathbb{R}_{\geq 0}\to\mathbb{R}^n$ be an essentially bounded signal and $\mathbf{E}\in\mathbb{R}^{n\times n}$ be a Hurwitz matrix.*

(a) If $\lim_{t\to\infty}\mathbf{u}(t)=\bar{\mathbf{u}}\in\mathbb{R}^n$, and $\mathbf{E}\in\mathbb{R}^{n\times n}$, then

$$\lim_{t\to\infty}\int_0^t \mathrm{e}^{\mathbf{E}\,(t-\tau)}\mathbf{u}(\tau)\mathrm{d}\tau=-\mathbf{E}^{-1}\,\bar{\mathbf{u}}. \tag{1.28}$$

(b) If $\lim_{t\to\infty}\int_0^t\mathbf{u}(\tau)\mathrm{d}\tau=\bar{\mathbf{u}}\in\mathbb{R}^n$, then

$$\lim_{t\to\infty}\int_0^t \mathrm{e}^{\mathbf{E}\,(t-\tau)}\mathbf{u}(\tau)\mathrm{d}\tau=\mathbf{0}. \tag{1.29}$$

Proof. To prove statement (a) we proceed as follows. Let $\boldsymbol{\mu}(t)=\mathbf{u}(t)-\bar{\mathbf{u}}$. Next, consider

$\dot{\boldsymbol{\zeta}} = \mathbf{E}\,\boldsymbol{\zeta} + \boldsymbol{\mu}$, $\boldsymbol{\zeta}(0) \in \mathbb{R}^n$, which gives $\boldsymbol{\zeta}(t) = \mathrm{e}^{\mathbf{E}\,t}\boldsymbol{\zeta}(0) + \int_0^t \mathrm{e}^{\mathbf{E}(t-\tau)}\boldsymbol{\mu}(\tau)\mathrm{d}\tau$, $t \geq 0$. Since $\mathbf{E}$ is Hurwitz and $\boldsymbol{\mu}$ is an essentially bounded and vanishing signal, by virtue of the ISS results for linear systems [88] we have $\lim_{t\to\infty} \boldsymbol{\zeta}(t) = 0$. Consequently, $\lim_{t\to\infty} \int_0^t \mathrm{e}^{\mathbf{E}\,(t-\tau)}\boldsymbol{\mu}(\tau)\mathrm{d}\tau = \mathbf{0}$, which guarantees (1.28).

To prove statement (b) we proceed as follows. Consider

$$\dot{\boldsymbol{\zeta}} = \mathbf{u},\ \dot{\boldsymbol{\eta}} = \mathbf{E}\boldsymbol{\eta} + \mathbf{u}, \quad \boldsymbol{\zeta}(0) = \mathbf{0},\ \boldsymbol{\eta}(0) \in \mathbb{R}^n,$$

which result in $\boldsymbol{\zeta}(t) = \int_0^t \mathbf{u}(\tau)\mathrm{d}\tau$ and

$$\boldsymbol{\eta}(t) = \mathrm{e}^{\mathbf{E}\,t}\boldsymbol{\eta}(0) + \int_0^t \mathrm{e}^{\mathbf{E}\,(t-\tau)}\mathbf{u}(\tau)\mathrm{d}\tau. \tag{1.30}$$

Given the conditions on $\mathbf{u}$ both $\boldsymbol{\zeta}$ and $\boldsymbol{\eta}$ are essentially bounded signals (recall that $\mathbf{E}$ is Hurwitz). Let $\mathbf{e} = \boldsymbol{\eta} - \boldsymbol{\zeta}$. Therefore, we can write

$$\dot{\mathbf{e}} = \mathbf{E}\,\mathbf{e} + \mathbf{E}\,\boldsymbol{\zeta}, \quad \mathbf{e}(0) = \boldsymbol{\eta}(0) \in \mathbb{R}^n.$$

Since $\boldsymbol{\zeta}$ is essentially bounded and satisfies $\lim_{t\to\infty} \mathbf{E}\boldsymbol{\zeta}(t) = \mathbf{E}\bar{\mathbf{u}}$, with an argument similar to that of the proof of statement (a), we can conclude that $\lim_{t\to\infty} \mathbf{e}(t) = -\bar{\mathbf{u}}$. As a result $\lim_{t\to\infty} \boldsymbol{\eta}(t) = \mathbf{0}$. Consequently, from (1.30), we obtain (1.29). □

Lemma 1.5.12. *Let $\mathcal{G}$ be a strongly connected and weight-balanced digraph. Then, every island of any agent i, is strongly connected and weight-balanced.*

Proof. Without loss of generality, we prove our argument by showing that the island $\mathcal{G}_1^1$ of agent 1 is strongly connected and weight-balanced. By construction, we know that there is a directed path from every agent to every other agent in $\mathcal{G}_1^1$, therefore, $\mathcal{G}_1^1$ is strongly connected. Next we show that $\mathcal{G}_1^1$ is weight-balanced. Let $\mathcal{V}_2 = \mathcal{V}_1^1 \backslash \{1\}$ and $\mathcal{V}_3 = \mathcal{V} \backslash \mathcal{V}_2$.Let the nodes of $\mathcal{G}$ be labeled in accordance to $(1, \mathcal{V}_2, \mathcal{V}_3)$, respectively, and partition the graph

Laplacian $\mathbf{L}$ accordingly as

$$\mathbf{L} = \begin{bmatrix} \mathsf{d}^1_{\text{out}} & -\mathbf{A}_{12} & -\mathbf{A}_{13} \\ -\mathbf{A}_{21} & \mathbf{L}_{22} & \mathbf{0} \\ -\mathbf{A}_{31} & \mathbf{0} & \mathbf{L}_{33} \end{bmatrix}.$$

Since $\mathcal{G}$ is strongly connected and weight-balanced, we have $\mathbf{L}\mathbf{1}_N = \mathbf{0}$ and $\mathbf{1}_N^\top \mathbf{L} = \mathbf{0}$, which guarantee that

$$\mathbf{1}^\top_{|\mathcal{V}_1^1|} \begin{bmatrix} -\mathbf{A}_{12} \\ \mathbf{L}_{22} \end{bmatrix} = \mathbf{0}, \qquad \begin{bmatrix} -\mathbf{A}_{21} & \mathbf{L}_{22} \end{bmatrix} \mathbf{1}_{|\mathcal{V}_1^1|} = \mathbf{0}. \tag{1.31}$$

Therefore,

$$\mathbf{1}^\top_{|\mathcal{V}_1^1|} \begin{bmatrix} -\mathbf{A}_{12} \\ \mathbf{L}_{22} \end{bmatrix} \mathbf{1}_{|\mathcal{V}_1^1|} = 0, \qquad\qquad \mathbf{1}^\top_{|\mathcal{V}_1^1|} \begin{bmatrix} -\mathbf{A}_{21} & \mathbf{L}_{22} \end{bmatrix} \mathbf{1}_{|\mathcal{V}_1^1|} = 0,$$

which we can use to conclude that $\mathtt{sum}(\mathbf{A}_{12}^\top) = \mathtt{sum}(\mathbf{A}_{21})$. Let the Laplacian matrix of $\mathcal{G}_1^1$ be $\mathbf{L}_1^1$. Partitioning this matrix according to order node set $(1, \mathcal{V}_2)$, we obtain

$$\mathbf{L}_1^1 = \begin{bmatrix} \mathsf{d}^{1,1}_{\text{out}} & -\mathbf{A}_{12} \\ -\mathbf{A}_{21} & \mathbf{L}_{22} \end{bmatrix},$$

where $\mathsf{d}^{1,1}_{\text{out}} = \sum_{j \in \mathcal{V}_2} \mathsf{a}_{1j} = \mathtt{sum}(\mathbf{A}_{12}^\top)$. To establish $\mathcal{G}_1^1$ is weight-balanced digraph, we show next that $\mathbf{1}^\top_{|\mathcal{V}_1^1|} \mathbf{L}_1^1 = \mathbf{0}$. From $\mathbf{1}_N^\top \mathbf{L} = \mathbf{0}$, it follows that $\mathbf{1}^\top_{|\mathcal{V}_1^1|} \begin{bmatrix} -\mathbf{A}_{12} \\ \mathbf{L}_{22} \end{bmatrix} = \mathbf{0}$. Therefore, to prove $\mathcal{G}_1^1$ is weight-balanced, we need to show that $\mathsf{d}^{1,1}_{\text{out}} + \mathtt{sum}(-\mathbf{A}_{21}) = 0$, which follows immediately from $\mathsf{d}^{1,1}_{\text{out}} = \mathtt{sum}(\mathbf{A}_{12}^\top)$ and $\mathtt{sum}(\mathbf{A}_{12}^\top) = \mathtt{sum}(\mathbf{A}_{21})$. □

1.5.6 Contraction Theory

Contraction theory assesses the stability properties of dynamical systems by studying the convergence behavior of neighboring trajectories [90]. The convergence is established by directly examining the evolution of the weighted Euclidean distance of close neighboring trajectories. Formally, consider a differentiable autonomous discrete-time dynamical system $g(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^n$ defined as

$$\mathbf{x}_{t+1} = g(\mathbf{x}_t), \tag{1.32}$$

with Jacobian

$$\nabla g(\mathbf{x}_t) = \left. \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}_t}. \tag{1.33}$$

Now, consider a differential displacement $\delta\mathbf{x}_t$. The differential displacement dynamics at $\mathbf{x}_t$ are governed by

$$\delta\mathbf{x}_{t+1} = \nabla g(\mathbf{x}_t)\delta\mathbf{x}_t. \tag{1.34}$$

The system dynamics $g(\mathbf{x}_t)$ are contractive if there exists a full rank state dependant metric $\boldsymbol{\Theta}(\mathbf{x}) \in \mathbb{R}^n \times \mathbb{R}^n$ such that the system trajectories satisfy

$$\|\boldsymbol{\Theta}(\mathbf{x}_t)\delta\mathbf{x}_t\| > \|\boldsymbol{\Theta}(\mathbf{x}_{t+1})\delta\mathbf{x}_{t+1}\|. \tag{1.35}$$

Equation (1.35) indicates that the weighted distance between any two infinitesimally close states decreases as the dynamics evolve [122]. When $\boldsymbol{\Theta}(\mathbf{x}) = \mathbf{I}$ the distances between trajectories are measured in the Euclidean norm sense. Figure 1.2 illustrates the behavior of two trajectories of a contractive system when $\boldsymbol{\Theta}(\mathbf{x}) = \mathbf{I}$.

For a small finite displacement $\Delta \mathbf{x}_t$, as an approximation of infinitesimal small displacement $\delta \mathbf{x}_t$, the first-order Taylor expansion of the system dynamics allows us to locally approximate the forward evolution of the displacement

$$\nabla g(\mathbf{x}_t)\Delta \mathbf{x}_t \approx g(\mathbf{x}_t + \Delta \mathbf{x}_t) - g(\mathbf{x}_t). \tag{1.36}$$

Thus, we may approximate the contraction condition (1.35) as

$$\|\boldsymbol{\Theta}(\mathbf{x}_{t+1})\left(g(\mathbf{x}_t + \Delta \mathbf{x}_t) - g(\mathbf{x}_t)\right)\| - \|\boldsymbol{\Theta}(\mathbf{x}_t)\Delta \mathbf{x}_t\| < 0. \tag{1.37}$$

Establishing a system as contractive allows for several useful stability properties to be deduced. We state motivating results from [90] in the following definition and proposition.

Definition 1. *Given the discrete-time system $\mathbf{x}_{t+1} = g(\mathbf{x}_t)$, a region of the state space is called a contraction region with respect to a uniformly positive definite metric $\mathbf{M}(\mathbf{x}_t) = \boldsymbol{\Theta}(\mathbf{x}_t)^T\boldsymbol{\Theta}(\mathbf{x}_t)$, if in that region*

$$\nabla g(\mathbf{x}_t)^T \mathbf{M}(\mathbf{x}_{t+1})\nabla g(\mathbf{x}_t) - \mathbf{M}(\mathbf{x}_t) < 0, \tag{1.38}$$

Proposition 1.5.3. *A convex contraction region contains at most one equilibrium point.*

It is shown in [90] that (1.38) is equivalent to condition (1.35) holding for all $\mathbf{x}_t$ in the contraction region. Thus, by Proposition 1.5.3, we may conclude that a unique equilibrium exists within a convex region if (1.35) holds everywhere inside the region. Therefore, (1.37) represents a useful numerical analog that can be enforced in order to drive a region towards being contractive. By choosing a set of $\Delta \mathbf{x}_t$, we will use condition (1.37) to enforce contracting behavior of the closed-loop system. Going beyond autonomous systems, when a system is subject to control input $\mathbf{u}_t$, i.e., $\mathbf{x}_{t+1} = g(\mathbf{x}_t) = f(\mathbf{x}_t, \mathbf{u}_t)$, contraction theory can be used to design state feedback policies $\mathbf{u}_t = \mathbf{u}(\mathbf{x}_t)$ such that the closed-loop system trajectories

converge to a given reference state. This may be done by determining $\mathbf{u}(\mathbf{x}_t)$ such that the convex region of interest is contractive and the unique equilibrium is the desired reference state. Such a control design process is outlined in the following sections.

1.6 Dissertation Outline

In chapter 2 to motivate our work, we study the problem of persistent monitoring of the finite number of inter-connected geographical nodes for event detection via a group of heterogeneous mobile agents. We tie a utility function to the maximum reward available in each point of interest and use it in our strategy selection algorithm to incentivize the agents to visit the geographical nodes with higher rewards. We formulate a general utility maximization problem as maximizing a submodular set function subject to partition matroid. In chapter 3 and 4 we then proceed to propose a suboptimal strategy selection algorithm with known optimality bound. We work in the value oracle model where the only access of the agents to the utility function is through a black box that returns the utility function value. The agents are communicating over a connected undirected graph and have access only to their own strategy set. In chapter 5, we design a distributed algorithm that enables each agent to find a suboptimal policy locally with a guaranteed level of privacy. In chapter 6, We particularly study the privacy preservation methods in consensus-based communication protocols. We study the problem of privacy preservation of the continuous-time Laplacian static average consensus algorithm using additive perturbation signals. In chapter 7, we proposed a method of certifying neural networks in the context of dynamical systems and policy-making using contraction theory.

Chapter 2

A Sub-modular Approach to Multi-agent Persistent Monitoring

In this chapter, We consider persistent monitoring of a finite number of inter-connected geographical nodes by a group of heterogeneous mobile agents. We assign to each geographical node a concave and increasing reward function that resets to zero after an agent's visit. Then, we design the optimal dispatch policy of which nodes to visit at what time and by what agent by finding a policy set that maximizes a utility that is defined as the total reward collected at visit times. We show that this optimization problem is NP-hard and its computational complexity increases exponentially with the number of the agents and the length of the mission horizon. By showing that the utility function is a monotone increasing and submodular set function of agents' policy, we propose a suboptimal dispatch policy design with a known optimality gap. To reduce the time complexity of constructing the feasible search set and also to induce robustness to changes in the operational factors, we perform our suboptimal policy design in a receding horizon fashion. Then, to compensate for the shortsightedness of the receding horizon approach we add a new term to our utility, which provides a measure of nodal importance beyond the receding horizon. This term gives the

policy design an intuition to steer the agents towards the nodes with higher rewards on the patrolling graph. Finally, we discuss how our proposed algorithm can be implemented in a decentralized manner. A simulation study demonstrates our results.

2.1 Problem Statement

We consider a persistent monitoring of a set of finite $\mathcal{V}$ inter-connected geographical nodes via a set of finite $\mathcal{A}$ mobile sensors/agents, where $|\mathcal{V}| > |\mathcal{A}|$. The mobile agents are confined to a set of pre-specified edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, e.g., aerial or ground corridors, to traverse from one node to another, see Fig. 2.1. Depending on their vehicle type, agents may have to take different edges to go from one node to another. Also, they may have different travel times along the same edge. We study dispatch policy that orchestrates the topological distribution of the mobile agents such that an optimized service for a global monitoring task is provided with a reasonable computational cost. To quantify the service objective we assign to each node $v \in \mathcal{V}$ the reward function,

$$R_v(t) = \begin{cases} 0, & t = \bar{t}_v, \\ \psi_v(t - \bar{t}_v), & t > \bar{t}_v, \end{cases} \tag{2.1}$$

where $\psi_v(t)$ is a nonnegative concave and increasing function of time and $\bar{t}_v$ is the latest time node v is visited by an agent. For example, in data harvesting or health monitoring, $\psi_v(.)$ can be the weighted idle time of the node v or in event detection, it can be the probability of at least one event taking place at inter-visit times.

To formalize our objective, we first introduce our notations and state our standing assumptions. For any node $v \in \mathcal{V}$, $\mathcal{N}_v$ is a set consisting node v and all the neighboring nodes that are connected to node v via an edge in $\mathcal{E}$. If there exists a path connecting node $v \in \mathcal{V}$ to

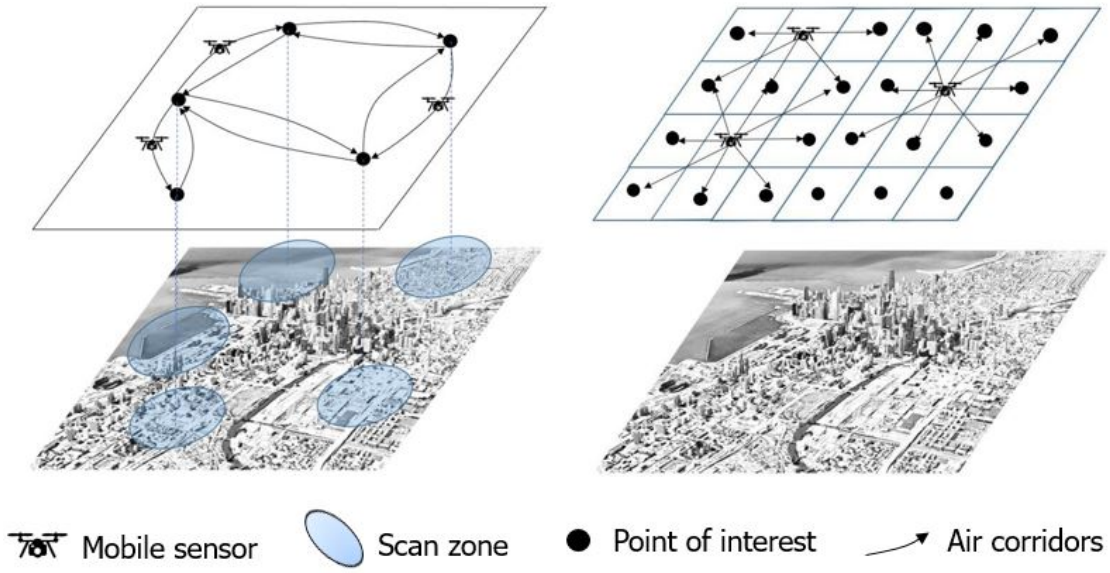


Figure 2.1 – Examples of a set of geographical nodes of interest and the edges between them. Finite number of nodes to monitor in a city can be restricted to some particular scanning zones (the picture on the left) or the cell partitioned map of the city (the picture on the right).

node $w \in \mathcal{V}$, we let $\tau^i_{v,w} \in \mathbb{R}_{>0}$ be the *shortest* travel time of agent $i \in \mathcal{A}$ from node v to w.

Assumption 1. *Upon arrival of any agent $i \in \mathcal{A}$ at any time $\bar{t} \in \mathbb{R}_{>0}$ at node $v \in \mathcal{V}$, the agent immediately scans the node and the reward $R_v(\bar{t})$ is scored for the patrolling team $\mathcal{A}$ and $\bar{t}_v$ of node v in (2.1) is set to $\bar{t}$. If more than one agent arrives at node $v \in \mathcal{V}$ and scans it at the same time $\bar{t}$, the reward collected for the team is still $R_v(\bar{t})$. If an agent $i \in \mathcal{A}$ needs to linger over each node for $\delta^i \in \mathbb{R}_{\geq 0}$ amount of time to complete its scan, during this time the agent cannot scan the node again to score a reward for the team.*

Let the tuple $\mathsf{p} = (\mathbf{V}_\mathsf{p}, \mathbf{T}_\mathsf{p}, \mathsf{a}_\mathsf{p})$ be a dispatch policy of agent $\mathsf{a}_\mathsf{p} \in \mathcal{A}$ over the given mission time horizon, where $\mathbf{V}_p$ and $\mathbf{T}_\mathsf{p}$ are the vectors that specify the nodes and the corresponding visit times assigned to agent a_p. Moreover, we let n_p be the total number of nodes visited by agent a_p, i.e., $\mathsf{n}_\mathsf{p} = \dim(\mathbf{V}_\mathsf{p})$. We refer to n_p as the *length* of the policy p. We refer to $(\mathbf{V}_\mathsf{p}(l), \mathbf{T}_\mathsf{p}(l))$, $l \in \{1, 2, \cdots, \mathsf{n}_\mathsf{p}\}$, as the l^{th} *step* of policy p. Furthermore, for any agent $i \in \mathcal{A}$, we let $\mathcal{P}^i$ be the set of all the admissible policies p over the mission horizon such that $\mathsf{a}_\mathsf{p} = i$.

Assumption 2. *For any policy* p*, we have* $\mathbf{V}_{\mathsf{p}}(l+1) \in \mathcal{N}_{\mathbf{V}_{\mathsf{p}}(l)}$*, for all* $l \in \{1, 2, \cdots, \mathsf{n}_{\mathsf{p}} - 1\}$.

We let $\mathcal{P} = \bigcup_{i \in \mathcal{A}} \mathcal{P}^i$. Then, given any $\bar{\mathcal{P}} \subset \mathcal{P}$, the utility function $\mathsf{R} : 2^{\mathcal{P}} \to \mathbb{R}_{>0}$ is $\bar{\mathcal{P}} \subset \mathcal{P}$, the utility function $\mathsf{R} : 2^{\mathcal{P}} \to \mathbb{R}_{>0}$ is

$$\mathsf{R}(\bar{\mathcal{P}}) = \sum\nolimits_{\forall \mathsf{p} \in \bar{\mathcal{P}}} \sum\nolimits_{l=1}^{\mathsf{n}_{\mathsf{p}}} R_{\mathbf{V}_{\mathsf{p}}(l)}(\mathbf{T}_{\mathsf{p}}(l)). \tag{2.2}$$

Given (2.2), the optimal policy to maximize the utility over a given mission horizon is given by

$$\mathcal{P}^{\star} = \underset{\bar{\mathcal{P}} \subset \mathcal{P}}{\operatorname{argmax}}\, \mathsf{R}(\bar{\mathcal{P}}), \quad \text{s.t.} \tag{2.3a}$$

$$|\bar{\mathcal{P}} \cap \mathcal{P}^i| \leq 1 \qquad i \in \mathcal{A}, \tag{2.3b}$$

where $|\,.\,|$ returns the cardinality of a set. The constraint condition (2.3b) is in the so-called *partition matroid* form [32] and restricts the choice of the optimal solution to be a set that contains of at most one member from each disjoint sets $\mathcal{P}^i,\ i \in \mathcal{A}$. A set value optimization problem of the form (2.3) is known to be NP-hard [123]. Lemma 2.1.1 below, whose proof is given in the appendix, gives the cost of constructing the feasible set $\mathcal{P}$ and time complexity of solving optimization problem (2.3).

Lemma 2.1.1 (Time complexity of problem (2.3a))**.** *The cost of constructing the feasible set* $\mathcal{P}$ *of optimization problem* (2.3a) *is of order* $O(\sum_{i \in \mathcal{A}} \mathsf{D}^{\bar{\mathsf{n}}^i})$*, where* $\mathsf{D} = \max_{v \in \mathcal{V}}(|\mathcal{N}_v|)$ *and* $\bar{\mathsf{n}}^i = \max\{\mathsf{n}_{\mathsf{p}}\}_{\forall \mathsf{p} \in \mathcal{P}^i}$*. Furthermore, the time complexity of solving optimization problem* (2.3a) *is* $O(\prod_{i \in \mathcal{A}} \mathsf{D}^{\bar{\mathsf{n}}^i})$.

Proof. The time complexity of constructing the admissible policy set $\mathcal{P}^i$ is of order of the number of possible paths that agent $i \in \mathcal{A}$ can traverse over the mission horizon while respecting Assumption 2, which is of order $\mathsf{D}^{\bar{\mathsf{n}}^i}$. Thus, the time complexity of constructing the feasible set $\mathcal{P} = \bigcup_{i \in \mathcal{A}} \mathcal{P}^i$ is $O(\sum_{i \in \mathcal{A}} \mathsf{D}^{\bar{\mathsf{n}}^i})$. Next, let $\bar{\mathcal{P}}$ be any subset of $\mathcal{P}$ that satisfies

constraint (2.3b). Due to Assumption 1, the reward scored by implementing policy $\mathsf{p} = (\mathbf{V}_\mathsf{p}, \mathbf{T}_\mathsf{p}, \mathsf{a}_\mathsf{p}) \in \bar{\mathcal{P}}$ cannot be calculated independent from the all the other policies in $\bar{\mathcal{P}} \backslash \{\mathsf{p}\}$. Hence, to solve optimization problem (2.3a), we need to evaluate all the possible policy sets $\bar{\mathcal{P}}$ satisfying the constraint (2.3b). Since $\bar{\mathcal{P}}$ can have at most one policy from the policy set $\mathcal{P}^i$ of $i \in \mathcal{A}$ and $\mathcal{P}^i$ has $O(\sum_{i\in\mathcal{A}} \mathsf{D}^{\bar{\mathsf{n}}^i})$ members, then $O(\prod_{i=1}^{M} \mathsf{D}^{\bar{\mathsf{n}}^i})$ different possibilities of $\bar{\mathcal{P}}$ exist which determines the time complexity of solving (2.3a). □

If the system parameters, such as number of the mobile agents or the nodes, or the parameters of $\psi_v(.)$ of the reward function at any node v, change after the optimal policy design, the optimization problem (2.3) should be solved again over the remainder of the mission horizon under the new conditions. Our objective in this chapter is to construct a suboptimal solution to solve the persistent monitoring problem given by (2.3) with polynomial time complexity. Moreover, we seek a solution that has intrinsic robustness to changes that can happen during the mission horizon.

We close this section by introducing some definitions and notations used subsequently. For any set function $g : 2^{\mathcal{Q}} \to \mathbb{R}$, we let

$$\Delta_g(\mathsf{q}|\bar{\mathcal{Q}}) = g(\bar{\mathcal{Q}} \cup \mathsf{q}) - g(\bar{\mathcal{Q}}),$$

for $\forall \bar{\mathcal{Q}} \in 2^{\mathcal{Q}}$ and $\forall \mathsf{q} \in \mathcal{Q}$, where Δ_g shows the increase in value of the set function g going from set $\bar{\mathcal{Q}}$ to $\bar{\mathcal{Q}} \cup \mathsf{q}$. Recall that $g : 2^{\mathcal{Q}} \to \mathbb{R}$ is *submodular* if and only if for two sets $\mathcal{Q}_1$ and $\mathcal{Q}_2$ satisfying $\mathcal{Q}_1 \subset \mathcal{Q}_2 \subset \mathcal{Q}$, and for $\mathsf{q} \notin \mathcal{Q}_2$ we have [32]

$$\Delta_g(\mathsf{q}|\bar{\mathcal{Q}}_1) \geq \Delta_g(\mathsf{q}|\bar{\mathcal{Q}}_2).$$

Then submodularity is a property of set functions that shows diminishing reward as new members are being introduced to the system. We say $g : 2^{\mathcal{Q}} \to \mathbb{R}$ is *monotone increasing* if

for all $\mathcal{Q}_1, \mathcal{Q}_2 \subset \mathcal{Q}$ we have $\mathcal{Q}_1 \subset \mathcal{Q}_2$ if and only if [32]

$$g(\mathcal{Q}_1) \leq g(\mathcal{Q}_2).$$

We denote a sequence of m real numbers $(\mathfrak{t}_1, \cdots, \mathfrak{t}_m)$ by $(\mathfrak{t})_1^m$. Given two increasing (resp. decreasing) sequences $(\mathfrak{t})_1^n$ and $(\mathfrak{v})_1^m$, $(\mathfrak{t})_1^n \oplus (\mathfrak{v})_1^m$ is their concatenated increasing (resp. decreasing) sequence, i.e., for $(\mathfrak{u})_1^{n+m} = (\mathfrak{t})_1^n \oplus (\mathfrak{v})_1^m$, any $\mathfrak{u}_k$, $k \in \{1, \cdots, n+m\}$ is either in $(\mathfrak{t})_1^n$ or $(\mathfrak{v})_1^m$ or is in both. We assume that $(\mathfrak{u})_1^{n+m}$ preserves the relative labeling of $(\mathfrak{t})_1^n$ or $(\mathfrak{v})_1^m$, i.e., if $\mathfrak{t}_k$ and $\mathfrak{t}_{k+1}$, $k \in \{1, \cdots, n-1\}$ (resp. $\mathfrak{v}_k$ and $\mathfrak{v}_{k+1}$, $k \in \{1, \cdots, m-1\}$) correspond to $\mathfrak{u}_i$ and $\mathfrak{u}_j$ in $(\mathfrak{u})_1^{n+m}$, then $i < j$.

2.2 Suboptimal Policy Design

According to Lemma 2.1.1 the time complexity of finding an optimal patrolling policy in (2.3a) increases exponentially by the maximum length, $\bar{\mathsf{n}}^i$, of the admissible policies of any agent $i \in \mathcal{A}$ and also by the number of the exploring agents M. In light of this observation, to reduce the computational cost, we propose the following suboptimal policy design. Since the maximum policy length $\bar{\mathsf{n}}^i$ is proportional to the length of the mission horizon, we first propose to trade in optimality and divide the planning horizon into multiple shorter horizons so that the policy design can be carried out in a consecutive manner over these shorter horizons. Then, to reduce the optimality gap and also to induce robustness to the online changes that can occur during the mission time, we propose to implement this approach in a receding horizon fashion where we calculate the policy over a specified shorter horizon but execute only some of the initial steps of the policy, and then we repeat the process. However, a receding horizon approach suffers from what we refer to as *shortsightedness*. That is, over large inter-connected geographical node sets, a receding horizon design is oblivious to the reward distribution of the nodes that are not in the feasible policy set in the planning horizon.

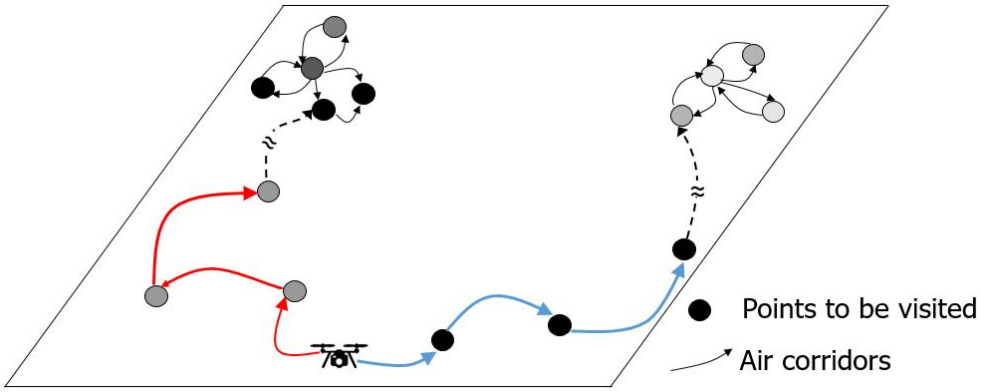


Figure 2.2 – An agent has two possible routes to take over the designated receding horizon. The nodes' color intensity shows their reward value. The blue route offers a higher reward over the receding horizon but it puts the agent close to an area with a lower amount of reward, while the red route results in lower total reward over the receding horizon but puts the agent near an area with higher amount of reward.

Then, the optimal policy over the planning horizon can inadvertently steer the agents away from the distant nodes with a higher reward, see Fig. 2.2. To compensate for this shortcoming, we introduce the notion of *nodal importance* and augment the reward function (2.2) over the design horizon with an additional term that given an admissible policy, provides a measure of how close an agent at the final step of the policy is to a cluster of geographical nodes with a high concentration of reward.

Let the augmented reward, whose exact form will be introduced below, over the planning horizon be $\bar{\mathsf{R}}$. Then, the optimal policy design over each receding horizon is

$$\mathcal{P}^\star = \underset{\bar{\mathcal{P}} \subset \mathcal{P}}{\operatorname{argmax}} \, \bar{\mathsf{R}}(\bar{\mathcal{P}}), \quad \text{s.t.} \quad |\bar{\mathcal{P}} \cap \mathcal{P}^i| \leq 1, \quad i \in \mathcal{A} \tag{2.4}$$

where hereafter $\mathcal{P} = \bigcup_{i \in \mathcal{A}} \mathcal{P}^i$ is the set of the union of the admissible policies of the agents $\mathcal{P}^i$, $i \in \mathcal{A}$, over the planning horizon. Hereafter, we let $\bar{\mathfrak{t}}_0^v$ be the last time node $v \in \mathcal{V}$ was visited before a planning horizon starts.

Next, to reduce the computational burden further, we propose to use Algorithm 1, which is a sequential greedy algorithm with a polynomial cost in terms of the number of the agents

Algorithm 1 Sequential Greedy Algorithm

procedure $\mathsf{SGOpt}(\mathcal{P}^i,\, i \in \mathcal{A})$
  Init: $\bar{\mathcal{P}} \leftarrow \emptyset,\ i \leftarrow 0,\ \{\bar{\mathfrak{t}}_v^0\}_{v\in\mathcal{V}}$
  for $i \in \mathcal{A}$ **do**
    $\mathsf{p}^{i\star} = \underset{\mathsf{p}\subset\mathcal{P}^i}{\operatorname{argmax}} \quad \Delta_{\bar{\mathsf{R}}}(\mathsf{p}|\bar{\mathcal{P}})$.
    $\bar{\mathcal{P}} \leftarrow \bar{\mathcal{P}} \cup \mathsf{p}^{i\star}$.
  end for
  Return $\bar{\mathcal{P}}$.
end procedure

to obtain a suboptimal solution for (2.4). In what follows, we show that since the objective function (2.4) is a submodular set function, Algorithm 1 comes with a known optimality gap. We also show that with a proper inter-agent communication coordination Algorithm 1 can be implemented in a decentralized manner.

For $v \in \mathcal{V}$, let $\mathcal{N}_v^r$ be the set consisted of node v itself and its r-hope neighbors. This set can be computed using the Breadth-first search in time $O(|\mathcal{E}| + |\mathcal{V}|)$ [124]. Here, $\tau_{w,v}^i$ can be computed via $A^\star$ algorithm in time $O(|\mathcal{E}|)$ [125]. Then, for every node $v \in \mathcal{V}$, we define the nodal importance with radius r at time τ as $L(v,\tau,r) = \sum_{w\in\mathcal{N}_v^r} R_w(\tau)$. Next, given an agent $i \in \mathcal{A}$ that is at node $w \in \mathcal{V}$ at time $\hat{t} \in \mathbb{R}_{\geq 0}$, we define the *relative nodal importance* of a node $v \in \mathcal{V}$ with respect to agent i as

$$\mathsf{L}(v,w,\hat{t},i) = L(v,\hat{t}+\tau_{w,v}^i,r)\big/\tau_{w,v}^i.$$

Then, $\mathsf{L}(v,\mathbf{V}_\mathsf{p}(\mathsf{n}_\mathsf{p}),\mathbf{T}_\mathsf{p}(\mathsf{n}_\mathsf{p}),\mathsf{a}_\mathsf{p})$ is a measure of the relative size of the awards concentration around any node $v \in \mathcal{V}$ that takes into account also the travel time of agent a_p from the final step of policy $\mathsf{p} = (\mathbf{V}_\mathsf{p},\mathbf{T}_\mathsf{p},\mathsf{a}_\mathsf{p}) \in \mathcal{P}$ to v. Let $\mathsf{L}(v,\mathsf{p})$ be the shorthand notation for $\mathsf{L}(v,\mathbf{V}_\mathsf{p}(\mathsf{n}_\mathsf{p}),\mathbf{T}_\mathsf{p}(\mathsf{n}_\mathsf{p}),\mathsf{a}_\mathsf{p})$. To compensate for the shortsightedness of the receding horizon design, then we revise the utility function to

$$\bar{\mathsf{R}}(\bar{\mathcal{P}}) = \mathsf{R}(\bar{\mathcal{P}}) + \alpha \sum\nolimits_{\forall \mathsf{p}\in\bar{\mathcal{P}}} \max_{\forall v\in\mathcal{V}} \mathsf{L}(v,\mathsf{p}), \quad \alpha \in \mathbb{R}_{\geq 0}. \tag{2.5}$$

The weighting factor $\alpha \in \mathbb{R}_{\geq 0}$ defines how much significance we want to assign to the distribution of the reward beyond the receding horizon. We should note that using a large α can gravitate the agents to move towards the nodes close to the anchor nodes, and make them oblivious to the rest of the nodes. For computational efficiency, instead of incorporating the relative nodal importance of all the nodes, which can be achieved by setting $\bar{\mathcal{V}}$ equal to $\mathcal{V}$, we propose to use only $\bar{\mathcal{V}}$ subset of the nodes. We refer to nodes in $\bar{\mathcal{V}}$ as *anchor nodes*. The anchor nodes can be selected to be the nodes with higher reward return or to be a set of nodes that are scattered uniformly on the graph. It is interesting to note that the relative nodal importance term in (2.5) is a reminiscent of terminal cost used in the model predictive control (MPC). In MPC, terminal cost that is used to achieve an infinite horizon control with closed-loop stability guarantees [126] in some way also compensates for the shortsightedness of the design over finite planning horizon. Next, we show that the reward function (2.5) is submodular over any given feasible policy set $\mathcal{P}$ in every planning horizon.

Theorem 2.2.1 (Submodularity of the reward function (2.5))**.** *For any weighting factor $\alpha \in \mathbb{R}_{\geq 0}$, the reward function $\bar{\mathsf{R}} : 2^{\mathcal{P}} \to \mathbb{R}_{>0}$ in* (2.5) *is a monotone increasing and submodular set function over $\mathcal{P}$.*

Proof. Let $\mathsf{c}(v, \mathcal{Q}) : \mathcal{V} \times 2^{\mathcal{Q}} \to \mathbb{Z}_{>0}$ be the total number of visits to the geographical node v, and $\mathcal{I}_{\mathcal{Q}} \subset \mathcal{V}$ be the set of the nodes that are visited when a policy set $\mathcal{Q} \subset \mathcal{P}$ is implemented. Furthermore, let the increasing sequence

$$(\mathfrak{t}^v(\mathcal{Q}))_1^{c(v,\mathcal{Q})} = (\mathfrak{t}_1^v(\mathcal{Q}), \mathfrak{t}_2^v(\mathcal{Q}), \cdots, \mathfrak{t}_{c(v,\mathcal{Q})}^v(\mathcal{Q}))$$

be the sequence of time that node $v \in \mathcal{I}_{\mathcal{Q}}$ was visited when agents implement $\mathcal{Q}$. Now consider the reward function $\bar{\mathsf{R}}$ in (2.5). Then, the first summand of $\bar{\mathsf{R}}$ expands as

$$\mathsf{R}(\bar{\mathcal{P}}) = \sum\nolimits_{v \in \mathcal{I}_{\bar{\mathcal{P}}}} \big(\sum\nolimits_{j=1}^{c(v,\bar{\mathcal{P}})} \psi_v(\Delta \mathfrak{t}_j^v(\bar{\mathcal{P}})) \big)$$

, where $\Delta\mathfrak{t}_j^v(\bar{\mathcal{P}}) = \mathfrak{t}_j^v(\bar{\mathcal{P}}) - \mathfrak{t}_{j-1}^v(\bar{\mathcal{P}})$ is the time between two consecutive visits of node v, and $\mathfrak{t}_0^v(\bar{\mathcal{P}}) = \bar{\mathfrak{t}}_0^v$. Next, consider the monitoring policy sets $\mathcal{Q}_1$, $\mathcal{Q}_2$ and monitoring policy q with $\mathcal{Q}_1 \subset \mathcal{Q}_2 \subset \mathcal{P}$, $\mathsf{q} \in \mathcal{P}$, $\mathsf{q} \notin \mathcal{Q}_1$, and $\mathsf{q} \notin \mathcal{Q}_2$. Because $(\mathfrak{t}^v(\mathcal{Q}_1))_1^{c(v,\mathcal{Q}_1)}$ is a sub-sequence of $(\mathfrak{t}^v(\mathcal{Q}_2))_1^{c(v,\mathcal{Q}_2)}$, using Lemma 1.5.2 and the fact that $\psi(.)_v$ is a normalized increasing concave function, we conclude that

$$\sum\nolimits_{j=1}^{c(v,\mathcal{Q}_1\cup\mathsf{q})} \psi_v(\Delta(\mathfrak{t}_j^v(\mathcal{Q}_2 \cup \mathsf{q}))) - \sum\nolimits_{j=1}^{c(v,\mathcal{Q}_1)} \psi_l(\Delta(\mathfrak{t}_j^v(\mathcal{Q}_2))) \geq 0$$

for $\forall v \in \mathcal{I}_{\bar{\mathcal{P}}}$. Therefore, $\Delta_{\mathsf{R}}(p|\mathcal{Q}_1) \geq 0$ which shows that $\mathsf{R}(\bar{\mathcal{P}})$ is a monotone increasing set function. Furthermore, using Lemma 1.5.3 we can write

$$\begin{aligned}
&\sum_{j=1}^{c(v,\mathcal{Q}_2\cup\mathsf{q})} \psi_v(\Delta(\mathfrak{t}_j^v(\mathcal{Q}_2 \cup \mathsf{q}))) - \sum_{j=1}^{c(v,\mathcal{Q}_2)} \psi_v(\Delta(\mathfrak{t}_j^v(\mathcal{Q}_2))) \\
&\qquad\qquad \leq \\
&\sum_{j=1}^{c(v,\mathcal{Q}_1\cup\mathsf{q})} \psi_v(\Delta(\mathfrak{t}_j^v(\mathcal{Q}_1 \cup \mathsf{q}))) - \sum_{j=1}^{c(v,\mathcal{Q}_1)} \psi_v(\Delta(\mathfrak{t}_j^v(\mathcal{Q}_1))).
\end{aligned}$$

Hence,

$$\Delta_{\mathsf{R}}(\mathsf{q}|\mathcal{Q}_1) \geq \Delta_{\mathsf{R}}(\mathsf{q}|\mathcal{Q}_2)$$

which shows that $\mathsf{R}(\bar{\mathcal{P}})$ is a submodular set function. Then, since the second summand of $\bar{\mathsf{R}}$, $\sum_{\forall \mathsf{p}\in\bar{\mathcal{P}}} \max_{\forall l\in\mathcal{V}} \mathsf{L}(l,\mathsf{p})$, is trivially positive and modular, the proof is concluded. □

Due to Theorem 2.2.1, the suboptimal dispatch policy of Algorithm 1, which has a polynomial computational complexity, has the following well-defined optimality gap.

Theorem 2.2.2 (Optimality gap of Algorithm 1)**.** *Let $\mathcal{P}^\star$ be an optimal solution of* (2.4) *and $\bar{\mathcal{P}}$ be the output of Algorithm 1. Then, $\bar{\mathsf{R}}(\bar{\mathcal{P}}) \geq \frac{1}{2}\bar{\mathsf{R}}(\mathcal{P}^\star)$.*

Proof. Since the objective function of (2.4) is monotone increasing and submodular over $\mathcal{P}$,

the proof follows by invoking [32, Theorem 5.1]. □

2.2.1 Comments on Decentralized Implementations of Algorithm 1

To implement Algorithm 1, given the current position of each agent and $\{\bar{\mathfrak{t}}_v^0\}_{v\in\mathcal{V}}$ at the beginning of each planning horizon, the admissible set of policies $\mathcal{P}^i$ for each agent $i \in \mathcal{A}$ should be calculated.

Let every agent know $\{\psi_v(t)\}_{v\in\mathcal{V}}$. A straightforward decentralized implement of Algorithm 1 then is a multi-centralized solution. In this solution, agents transmit the feasible policy sets across the entire network until each agent knows the whole policy set $\mathcal{P}^i$, $\forall i \in \mathcal{A}$ (flooding approach). Then, each agent acts as a central node and runs a copy of Algorithm 1 locally. Although reasonable for small-size networks, the communication and storage costs of this approach scale poorly with the network size. The sequential structure of Algorithm 1 however, offers an opportunity for a communicationally and computationally more efficient decentralized implementations, as described in steps 1 to 9 of Algorithm 2. Step 10 of Algorithm 2 is included for receding horizon implementation purpose, where the execution plan can be for example one or all of the agents visit at least one node. To implement Algorithm 2, we assume that the agents $\mathcal{A}$ can form a bidirectional connected communication graph $\mathcal{G}^a = (\mathcal{A}, \mathcal{E}^a)$, i.e., there is a path from every agent to every other agent on $\mathcal{G}^a$. Then, there always exists a route $\texttt{SEQ} = \mathtt{s}_1 \to \cdots \to \mathtt{s}_i \to \cdots \to \mathtt{s}_K$, $\mathtt{s}_k \in \mathcal{A}$, $k \in \{1, \cdots, K\}$, $K \geq M$, that visits all the agents (not necessarily only one time), see Fig. 2.3(a). The agents follow SEQ to share their information while implementing Algorithm 2. The communication cost to execute Algorithm 2 can be optimized by picking SEQ to be the shortest path [127] that visits all the agents over graph $\mathcal{G}^a$. If $\mathcal{G}^a$ has a Hamiltonian path, the optimal choice for SEQ is a Hamiltonian path. Recall that a Hamiltonian path is a path that visits every agent on $\mathcal{G}^a$ only once [128]. When, there is a SEQ that visits every agent on $\mathcal{G}^a$, the directed

Algorithm 2 Decentralized Implementation of Sequential Greedy Algorithm

1: Init: $\bar{\mathcal{P}} \leftarrow \emptyset,\ i \leftarrow 1,\ \{\bar{\mathfrak{t}}_v^0\}_{v\in\mathcal{V}}$
2: **while** $i \leq K$ **do**
3: **if** $\mathtt{s}_i$ is being called for the first time **then**
4: agent $\mathtt{s}_i$ computes $\mathsf{p}^{\mathtt{s}_i\star} = \underset{\mathsf{p}\subset\mathcal{P}^{\mathtt{s}_i}}{\operatorname{argmax}} \quad \Delta_{\bar{\mathsf{R}}}(\mathsf{p}|\bar{\mathcal{P}})$.
5: $\bar{\mathcal{P}} \leftarrow \bar{\mathcal{P}} \cup \mathsf{p}^{\mathtt{s}_i\star}$.
6: **end if**
7: agent $\mathtt{s}_i$ pass $\bar{\mathcal{P}}$ to $\mathtt{s}_{i+1}$.
8: $i \leftarrow i+1$.
9: **end while**
10: agent $\mathtt{s}_K$ based on the execution plan of the receding horizon operation updates $\{\bar{\mathfrak{t}}_v^0\}_{v\in\mathcal{V}}$ and communicates it to the team

information graph $\mathcal{G}^I = (\mathcal{A}, \mathcal{E}^I)$ of Algorithm 2, which shows the information access of each agent while implementing Algorithm 2, is full, see Fig. 2.3. That is, each agent in `SEQ` is aware of the previous agents' decision. Therefore, the solution obtained by Algorithm 2 is an exact sequential greedy algorithm and its optimality gap is 1/2. We recall that the labeling order of the mobile agents does not have an effect on the optimality gap guaranteed by Theorem 2.2.2 [129]. If an agent $i \in \mathcal{A}$ appears repeatedly in `SEQ` (e.g., the blue agent in Fig. 2.3), with a slight increase in computation cost, we can modify Algorithm 2 to allow agent i to redesign and improve its sub-optimal policy $\mathsf{p}^{i\star}$ by re-executing step 4 of Algorithm 2.

Another form of decentralized implementation of Algorithm 1, which may be more relevant in urban environments, is through a client-server framework implemented over a cloud. In this framework, agents (clients) connect to shared memory on a cloud (server) to download or upload information or use the cloud's computing power asynchronously. Let $\{\mathcal{T}^i\}$, $i \in \mathcal{A}$, be the set of disjoint time slots that is allotted respectively to agents $\mathcal{A}$, see Fig. 2.4. To implement Algorithm 1, agent $i \in \mathcal{A}$ connects to the server at the beginning of $\mathcal{T}^i$ to check out $\bar{\mathcal{P}}$ and $\{\bar{\mathfrak{t}}_v^0\}_{v\in\mathcal{V}}$. Then, it completes steps 4 and 5 of Algorithm 1, and checks in the updated $\bar{\mathcal{P}}$ to the server before $\mathcal{T}^i$ elapses fully. The last agent based on the execution plan of the receding horizon operation updates $\{\bar{\mathfrak{t}}_v^0\}_{v\in\mathcal{V}}$ and checks it in the cloud memory for

Figure 2.3 – The plot on the left shows the bi-directional communication graph $\mathcal{G}^a$ in black along with an example `SEQ` path in red. The plot on the right shows the complete information sharing graph $\mathcal{G}^I$ if agents follow `SEQ` while implementing Algorithm 2. Arrow going from agent i to agent j means that agent j receives agent i's information.

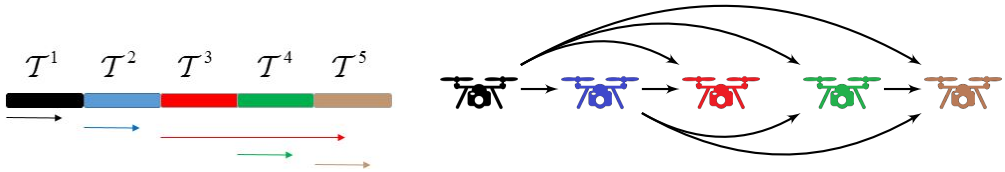


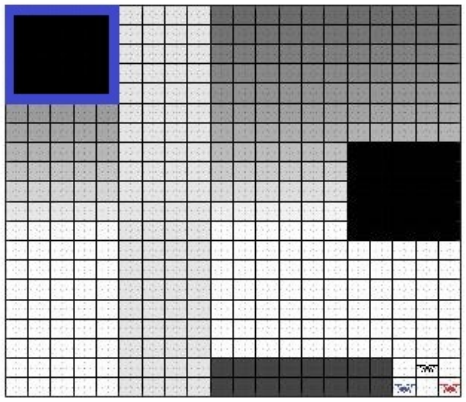
Figure 2.4 – $\{\mathcal{T}^i\}_{i\in\mathcal{A}}$, $\mathcal{A}=\{1,2,3,4,5\}$ are the time slots allotted to each agent to connect to the cloud. The arrows show the time each agent took to do their calculations for an example scenario. Here, the associated information graph $\mathcal{G}^I$ is as the incomplete graph on the right with clique number of 3.

next receding horizon planning. Since the time slots assigned to the agents do not overlap, agent i has access to policy $\mathsf{p}^{k\star}$ of all agents k which has already communicated to the cloud. Thus, the information graph $\mathcal{G}^I$ is full, and the optimality gap of 1/2 holds.

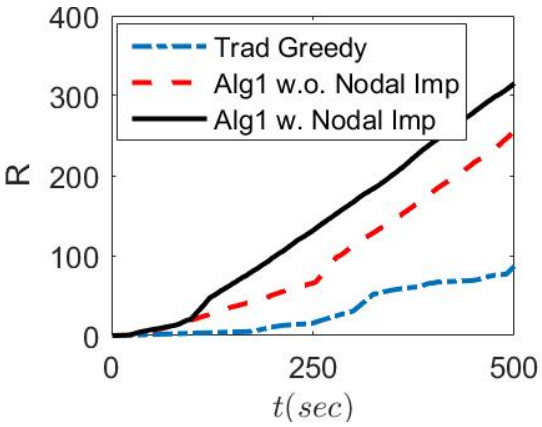
If there is a message dropout while executing Algorithm 2 or in the decentralized server-client based operation an agent j takes a longer time than $\mathcal{T}^j$ to complete and check-in $\bar{\mathcal{P}}$ to the cloud, the information graph becomes incomplete, see for example Fig. 2.4. Then, the corresponding decentralized implementation deviates from the exact sequential greedy Algorithm 2. For such cases, [129] shows that the optimality gap instead of 1/2 becomes $\frac{1}{M-\omega(\mathcal{G}^I)+2}$, where $\omega(\mathcal{G}^I)$ is the clique number of $\mathcal{G}^I$ [129]. Recall that the clique number of a graph is equal to the number of the nodes in the largest sub-graph such that adding an edge will cause a cycle [130].

2.3 Numerical Evaluations

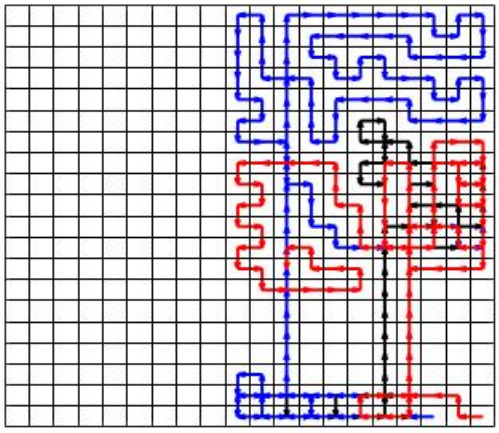
We consider persistent monitoring using 3 agents for event detection over an area that is divided into 20 by 20 grid map as shown in Fig. 2.5(a). The geographical nodes of interest $\mathcal{V}$ are the center of the cells in Fig. 2.5(a). The agents can travel from a cell to the neighboring cells in the right, left, bottom, and top. The agents are homogeneous and the travel time between any neighboring nodes for all the agents are identical and equal to 1 second. The agents start their patrolling task from the nodes where they are depicted in Fig. 2.5(a). We model the event occurrence in each geographical node as a Poisson process and define our reward function at each node $v \in \mathcal{V}$ as (2.1) with $\psi_v(t) = 1 - \mathrm{e}^{\lambda_v t}$ where $\lambda_v \in \mathbb{R}_{>0}$ is the arrival rate of the event; for more details see [131]. Fig. 2.5(a) shows the reward value of the nodes at $t = 120$ seconds when there is no monitoring. The color intensity of the cells in Fig. 2.5(a) is proportional to λ_v; the higher λ_v, the darker the color of node v. The region enclosed by the blue rectangle initially has a low reward but after 100 seconds its reward value is increased to a higher value by changing λ_v of the corresponding cells. An animated depiction of the change in the reward map because of different dispatch policies we discuss below is available in [132]. We compare the performance of Algorithm 1, implemented in a receding horizon fashion, and a conventional greedy algorithm where each agent always moves to the neighboring node that has the instantaneous highest reward value. In implementing Algorithm 1 in a receding horizon fashion, we assume that the planning horizon is 4 seconds and the execution horizon is 1 second. We consider both the case of including ($\alpha = 0.1$) and excluding ($\alpha = 0$) the nodal importance measure in the reward function (2.5). Fig. 2.5(b) shows that the traditional greedy cell selection performs poorly compared to the other two planning algorithms. The reason is that the three agents' decision becomes the same after a while, i.e., they start choosing the same cell after a while and moving together, therefore all three agents act as if one agent is patrolling (recall Assumption 1). The performance of Algorithm 1 is better than a standard greedy cell selection because the effect of agent i's



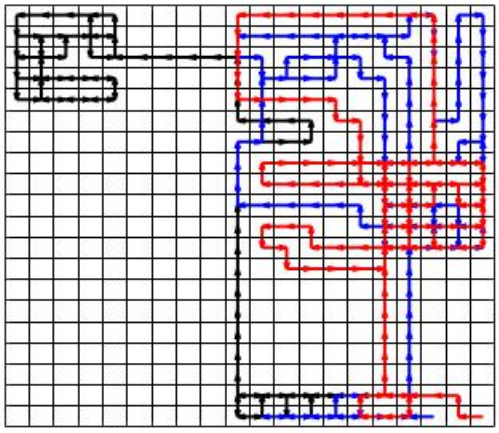
(a) Reward map



(b) The collected reward



(c) Agents' path when they follow Algorithm 1 and use $\alpha = 0$ over $[0, 150]$ seconds



(d) Agents' path when they follow Algorithm 1 and use $\alpha = 0.1$ over $[0, 150]$ seconds

Figure 2.5 – Three agents patrol a field, divided into 20 by 20 cells.

patrolling policy is taken into account when agent $i + 1$ is designed. Therefore, the chances that all three agents go to the same cell together and move together is narrow. Furthermore, we can note that implementing Algorithm 1 by considering the effect of nodal importance delivers a better outcome. The reason is that in the case that there is no nodal importance, the agents are drawn to the region of high importance near them and stay there as Fig. 2.5(c) shows. However, there are other important regions with higher values that are farther away, especially the area on the left top corner which is separated by a low rate stripe from where agents start. Incorporating nodal importance, as Fig. 2.5(d) shows steers the agents to the regions with a higher rate of reward that are beyond the receding horizon's sight.

2.4 Conclusions

We presented a multi-agent dispatch policy design for persistent monitoring of a set of finite inter-connected geographical nodes. Our design relied on assigning an increasing and concave reward function of time to each node that reset to zero after a visit by an agent. We defined our design utility function as the sum of the rewards scored for the team when agents visit the geographical nodes. By showing that the utility function is a monotone increasing and submodular set function, we laid the ground to propose a suboptimal solution with a known optimality gap for our dispatch policy design, which was NP-hard. To induce robustness to the changes in the problem parameters, we proposed our suboptimal solution in a receding horizon setting. Next, to compensate for the shortsightedness of the receding horizon approach, we added a new term, called the relative nodal importance, to the utility function as a measure to incorporate a notion of the importance of the regions beyond the feasible solution set of the receding horizon optimization problem. Our numerical example demonstrated the benefit of introducing this term. Lastly, we discussed how our suboptimal solution can be implemented in a decentralized manner. Our future work is to investigate decentralized algorithms that allow agents to communicate synchronously with each other in order to have a consensus on a policy with a known optimality gap.

Chapter 3

Distributed Strategy Selection I

Constrained submodular set function maximization problems often appear in multi-agent decision-making problems with a discrete feasible set. A prominent example is the problem of multi-agent mobile sensor placement over a discrete domain. However, submodular set function optimization problems are known to be NP-hard. In this chapter, we consider a class of submodular optimization problems that consists of maximization of a monotone and submodular set function subject to a uniform matroid constraint over a group of networked agents that communicate over a connected undirected graph. Our objective is to obtain a distributed suboptimal polynomial-time algorithm that enables each agent to obtain its respective policy via local interactions with its neighboring agents. Our solution is a fully distributed gradient-based algorithm using the multilinear extension of the submodular set functions and exploiting a maximum consensus scheme. This algorithm results in a policy set that when the team objective function is evaluated at worst case the objective function value is in $1 - 1/\mathrm{e} - O(1/T)$ of the optimal solution. An example demonstrates our results.

3.1 Problem Statement

We consider a group of $\mathcal{A} = \{1, ..., N\}$ with communication and computation capabilities interacting over a connected undirected graph $\mathcal{G}(\mathcal{A}, \mathcal{E})$. These agents aim to solve in a distributed manner the optimization problem

$$\max_{\mathcal{R} \in \mathcal{I}} f(\mathcal{R}) \quad \text{s.t.} \tag{3.1a}$$

$$\mathcal{I} = \big\{ \mathcal{R} \subset \mathcal{P} \,\big|\, |\mathcal{R} \cap \mathcal{P}_i| \leq 1,\ i \in \mathcal{A} \big\}, \tag{3.1b}$$

where utility function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ is monotone increasing and submodular set function over the discrete policy space $\mathcal{P} = \bigcup_{i=1}^{N} \mathcal{P}_i$, with $\mathcal{P}_i$ being the policy space of agent $i \in \mathcal{A}$, which is only known to agent i. In this chapter, we work in the value oracle model where the only access to the utility function is through a black box returning $f(\mathcal{R})$ for a given set $\mathcal{R}$. Every agent can obtain the value of the utility function at any subset $\mathcal{R} \in \mathcal{P}$. The constraint (4.1b) is the partitioned matroid $\mathcal{M} = \{\mathcal{P}, \mathcal{I}\}$, which ensures that only one policy per agent is selected from each local policy set $\mathcal{P}_i$, $i \in \mathcal{A}$. An example application scenario of our problem of interest is shown in Fig. 3.2. Without loss of generality, to simplify notation, we assume that the policy space is $\mathcal{P} = \{1, ..., n\}$, and is sorted agent-wise with $1 \in \mathcal{P}_1$ and $n \in \mathcal{P}_N$.

Finding the optimal solution $\mathcal{R}^\star \in \mathcal{I}$ of (3.1) even in central form is NP-Hard [45]. The computational time of finding the optimizer set increases exponentially with N [131]. The well-known sequential greedy algorithm finds a suboptimal solution $\mathcal{R}_{\text{SG}}$ for (3.1) with the optimality bound of $f(\mathcal{R}_{\text{SG}}) \geq \frac{1}{2} f(\mathcal{R}^\star)$, i.e., a 1/2-approximation at worst case [45]. More recently, by using the multilinear extension of the submodular utility functions, Vondrák [4] developed a randomized centralized continuous greedy algorithm which achieves a $(1-1/\text{e})-O(1/T)$-approximate for set value optimization (3.1) in the value oracle model, see Algo-

Algorithm 3 Practical implementation of the continuous greedy process [4].

$\mathbf{x} \leftarrow \mathbf{0}$, Init: $t \leftarrow 1$,
while $t \leq T$ **do**
  Draw K samples of $\mathcal{R}$ from $\mathcal{P}$ according to $\mathbf{x}$
  for $p \in \mathcal{P}$ **do**
    Estimate $w_p \sim \mathbb{E}[f(\mathcal{R}_{\mathbf{x}} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}} \setminus \{p\})]$
  end for
  Solve for $\mathcal{R}^\star = \underset{\mathcal{R} \in \mathcal{I}}{\operatorname{argmax}} \sum_{p \in \mathcal{R}} w_p$.
  Update membership vector as $\mathbf{x} \leftarrow \mathbf{x} + \frac{1}{T}\mathbf{1}_{\mathcal{R}^\star}$
  $t \leftarrow t + 1$
end while
Use Pipage rounding to convert the fractional solution $\mathbf{x}$ to an integral solution.

Note: $\mathbf{1}_{\mathcal{R}^\star}$ is the $\mathbf{v}$ in (3.3).

rithm 3[1]. Our objective in this chapter is to develop a distributed implementation of Algorithm 3 to solve (3.1) for when agents interact over a connected graph $\mathcal{G}$. Recall that in our problem setting, every agent $i \in \mathcal{A}$ can evaluate the utility function for a given $\mathcal{R} \subset \mathcal{P}$ but it has access only to its own policy space $\mathcal{P}_i$.

3.2 A Polynomial-Time Distributed Multi-Agent Randomized Continuous Greedy Algorithm

Our proposed distributed multi-agent randomized continuous greedy algorithm to solve the set value optimization problem (3.1) over a connected graph $\mathcal{G}$ is given in Algorithm 4, whose convergence guarantee and suboptimality gap is given in Theorem 3.2.2 below. To provide the insight into construction of Algorithm 4, we first review the idea behind the central suboptimal solution via Algorithm 3 following [4]. We also provide some intermediate results that will be useful in establishing our results.

[1]Pipage rounding is a polynomial time algorithm which moves a fractional point $\mathbf{x}$ on a hypercube to a integral point $\hat{\mathbf{x}}$ on the same hypercube such that $f(\hat{\mathbf{x}}) \geq f(\mathbf{x})$

3.2.1 A Short Overview Of The Central Continuous Greedy Process

As we mentioned earlier the continuous multilinear extension is a relaxation strategy that extends a submodular function $f(\mathcal{R})$, which is defined on the vertices of the $n-$dimensional hypercube $\{0,1\}^n$ to a continuous multilinear function F defined on $[0,1]^n$. The two functions evaluate identically for any vector $\mathbf{x} \in [0,1]^n$ that is the membership indicator vector of a set $\mathcal{R} \in \mathcal{P}$. Then, by way of a process that runs continuously, depending only on local properties of F, we can produce a point $\mathbf{x} \in P(\mathcal{M})$ to approximate the optimum $OPT = \max_{\mathbf{x} \in P(\mathcal{M})} F(\mathbf{x})$ (here recall (4.6)). The proposal is to move in the direction of a vector constrained by $P(\mathcal{M})$ which maximizes the local gain. To understand the logic behind Algorithm 3 let us review the conceptual steps of this continuous greedy process.Lets views the process as a particle starting at $\mathbf{x}(0) = \mathbf{0}$ and following the flow

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}) \quad \text{where} \quad \mathbf{v}(\mathbf{x}) = \arg\max_{\mathbf{v} \in P(\mathcal{M})} (\mathbf{v}.\nabla F(\mathbf{x})). \tag{3.2}$$

over a unit time interval $[0,1]$. We note that $\mathbf{x}(t)$ for $t \in [0,1]$ is contained in $P(\mathcal{M})$, since it is a convex combination of vectors in $P(\mathcal{M})$.

Lemma 3.2.1. *Consider the set value optimization problem* (3.1). *Suppose* $f : \mathcal{P} \to \mathbb{R}_{\geq 0}$ *is an increasing and submodular set function and consider its multilinear extension* $F : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$. *Then* $\forall \mathbf{x} \in P(\mathcal{M})$

$$\mathbf{1}_{\mathcal{S}^\star}.\nabla F(\mathbf{x}) \geq f(\mathcal{S}^\star) - F(\mathbf{x}).$$

and

$$\frac{dF}{dt} = \mathbf{v}(\mathbf{x}).\nabla F(\mathbf{x}) \geq f(\mathcal{S}^\star) - F(\mathbf{x}).$$

Proof. Consider a point $\mathbf{x}$ along the trajectory of our flow (3.2) and assume that $\mathbf{x}^\star$ is the true optimum $OPT = F(\mathbf{x}^\star) = f(\mathcal{S}^\star)$. Now consider a direction $\mathbf{v}^\star = \max\{\mathbf{x}^\star - \mathbf{x}, \mathbf{0}\}$, which is a nonnegative vector. Because $0 \leq \mathbf{v}^\star \leq \mathbf{x}^\star \in P(\mathcal{M})$, then $\mathbf{v}^\star \in P(\mathcal{M})$. By virtue of Lemma 1.5.5, F is monotone increasing, therefore, using $\max\{\mathbf{x}^\star - \mathbf{x}, \mathbf{0}\} = \max\{\mathbf{x}^\star, \mathbf{x}\} - \mathbf{x}$ we have $F(\mathbf{x} + \mathbf{v}^\star) = F(\max\{\mathbf{x}^\star, \mathbf{x}\}) \geq F(\mathbf{x}^\star)$. However, $\mathbf{x} + \mathbf{v}^\star$ does not belong to $P(\mathcal{M})$ necessarily. Therefore, lets consider $F(\mathbf{x}+\zeta\mathbf{v}^\star)$ for $\zeta \geq 0$. It can be shown that $F(\mathbf{x}+\zeta\mathbf{v}^\star)$ is concave in ζ and $\frac{\mathrm{d}F}{\mathrm{d}\zeta}$ is non-increasing. Thus, it can be established that $F(\mathbf{x}+\mathbf{v}^\star) - F(\mathbf{x}) \leq \frac{\mathrm{d}F}{\mathrm{d}\zeta}|_{\zeta=0} = \mathbf{v}^\star.\nabla F(\mathbf{x})$. But, since $\mathbf{v}^\star \in P(\mathcal{M})$, and $\mathbf{v} \in P(\mathcal{M})$ that is used to generate $\mathbf{v}$ maximizes $\mathbf{v}.\nabla F(\mathbf{x})$, we can write $\mathbf{v}.\nabla F(\mathbf{x}) \geq \mathbf{v}^\star.\nabla F(\mathbf{x}) \geq F(\mathbf{x}+\mathbf{v}^\star) - F(\mathbf{x}) \geq OPT - F(\mathbf{x})$. Now we note that $\frac{\mathrm{d}F}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}).\nabla F(\mathbf{x}) \geq OPT - F(\mathbf{x}(t))$. Moreover, since $\mathbf{1}_{\mathcal{S}^\star}.\nabla F(\mathbf{x}) \leq \mathbf{v}^\star.\nabla F(\mathbf{x})$ then we have $\mathbf{1}_{\mathcal{S}^\star}.\nabla F(\mathbf{x}) \geq f(\mathcal{S}^\star) - F(\mathbf{x})$. □

Therefore, given $\mathbf{x}(0) = \mathbf{0}$, using the Comparison Lemma [107], we can conclude the $F(\mathbf{x}) \geq (1-\mathrm{e}^{-t})OPT$, and thus $\mathbf{x}(1) \in P(\mathcal{M})$ and also $F(\mathbf{x}(1)) \geq (1-1/\mathrm{e})OPT$. In the second stage of the algorithm, the fractional solution $\mathbf{x}(1)$ is rounded to a point in $\{0,1\}^n$ by use of Pipage rounding method, see [60] for more details about Pipage rounding. The aforementioned exposition is the conceptual design behind the continuous greedy process. The algorithm 3is a practical implementation achieved by the use of a numerical iterative process

$$\mathbf{x}(t+1) = \mathbf{x}(t) + \frac{1}{T}\mathbf{v}(t), \tag{3.3}$$

and use of sampling to compute $\nabla F(\mathbf{x})$ and consequently $\mathbf{v}(t)$, see [4] for more details. In what follows, we explain a practical distributed implementation of the the continuous greedy process, which is realized as Algorithm 4, and is inspired by this central solution.

Algorithm 4 Discrete Distributed implementation of the continuous greedy process.

Init: $\bar{\mathcal{P}} \leftarrow \emptyset,\ \mathcal{F}_i \leftarrow \emptyset,\ t \leftarrow 1,$
while $t \leq T$ **do**
  for $i \in \mathcal{A}$ **do**
    Draw K_i sample policy sets $\mathcal{R}$ such that $q \in \mathcal{R}$ with the probability α for all $(q, \alpha) \in \mathcal{F}_i$.
    for $p \in \mathcal{P}_i$ **do**
      Compute $w_p^i \sim \mathbb{E}[f(\mathcal{R} \cup \{p\}) - f(\mathcal{R} \setminus \{p\})]$ using the policy sample sets of step 4.
    end for
    Solve for $p^\star = \underset{p \in \mathcal{P}_i}{\operatorname{argmax}}\ w_p^i$.
    $\mathcal{F}_i^- \leftarrow \mathcal{F}_i \oplus \{(p^\star, \frac{1}{T})\}$.
    Broadcast $\mathcal{F}_i^-$ to the neighbors $\mathcal{N}_i$.
    $\mathcal{F}_i \leftarrow \underset{j \in \mathcal{N}_i \cup \{i\}}{\operatorname{MAX}}\ \mathcal{F}_j^-$
  end for
  $t \leftarrow t + 1$.
end while
for $i \in \mathcal{A}$ **do**
  Sample one policy $\bar{p} \in \mathcal{P}_i$ using $\mathcal{F}_i$
  $\bar{\mathcal{P}} \leftarrow \bar{\mathcal{P}} \cup \{\bar{p}\}$
end for

3.2.2 Design and Analysis of the Distributed Continuous Greedy Process

We start off our design and analysis of the distributed continuous greedy process by introducing our notation and the set of computations that agents carry out locally using their local information and interaction with their neighbors. The algorithm is performed over the finite time horizon of T steps. Let $\mathcal{F}_i(t) \subset \mathcal{P} \times [0, 1]$ be the information set of agent i at time step t, initialized at $\mathcal{F}_i(0) = \emptyset$. For, any couple $(p, \alpha) \in \mathcal{F}_i(t)$ the first element is the policy and the second element is the corresponding membership probability. We let $\mathbf{x}_i(t) \in \mathbb{R}^n$ (recall $|\mathcal{P}| = n$) be the local copy of the membership probability of our suboptimal solution of (3.1) at agent i at time step t, defined according to

$$x_{ip}(t) = \begin{cases} \alpha, & (p, \alpha) \in \mathcal{F}_i(t), \\ 0 & \text{otherwise}, \end{cases} \qquad p \in \mathcal{P}. \tag{3.4}$$

Recall that $\mathcal{P} = \{1, ..., n\}$ and it is sorted agent-wise with $1 \in \mathcal{P}_1$ and $n \in \mathcal{P}_N$. Hence, $\mathbf{x}_i(t) = [\hat{\mathbf{x}}_{i1}^\top(t), \cdots, \mathbf{x}_{ii}^\top(t), \cdots, \hat{\mathbf{x}}_{iN}^\top(t)]^\top$ where $\mathbf{x}_{ii}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_i|}$ is the membership probability vector of agent i's own policy at iteration t, while $\hat{\mathbf{x}}_{ij}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_j|}$ is the local estimate of the membership probability vector of agent j by agent i. At time step t agent i solves the optimization problem

$$\widetilde{\mathbf{v}}_i(t) = \underset{\mathbf{y} \in P_i(\mathcal{M})}{\operatorname{argmax}} \mathbf{y}.\widetilde{\nabla F}(\mathbf{x}_i(t)) \tag{3.5}$$

with

$$P_i(\mathcal{M}) = \Big\{ [\mathbf{y}_1^\top, \cdots, \mathbf{y}_N^\top]^\top \in \mathbb{R}_{\geq 0}^n \;\Big|\; \mathbf{1}^\top.\mathbf{y}_i \leq 1, \quad \mathbf{y}_j = \mathbf{0},\ j \in \mathcal{A} \backslash \{i\} \Big\}. \tag{3.6}$$

The term $\widetilde{\nabla F}(\mathbf{x}_i(t))$ is the estimate of $\nabla F(\mathbf{x}_i(t))$ which is calculated by taking K_i samples of set $\mathcal{R}_{\mathbf{x}_i(t)}$ according to membership probability vector $\mathbf{x}_i(t)$. Recall (4.4). Hence, $\frac{\partial F}{\partial x_p}$ is estimated by averaging $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\})$ over the samples. We denote the pth element of $\widetilde{\nabla F}(\mathbf{x}_i(t))$ by w_p^i, and represent it by

$$w_p^i \sim \mathbb{E}[f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\})]. \tag{3.7}$$

We note that given the definition of $P_i(\mathcal{M})$, to compute (3.5), we only need w_p^i for $p \in \mathcal{P}_i$.

Remark 3.2.1 (Local computation of w_p^i, $p \in \mathcal{P}_i$, by agent i)**.** *Given the definition* (3.4)*, we note that w_p^i, an estimate of $\frac{\partial F}{\partial x_p}$ can be obtained from drawing K_i sample policy sets $\mathcal{R}$ such that $q \in \mathcal{R}$ with the probability α for all $(q, \alpha) \in \mathcal{F}_i$ and using*

$$w_p^i \sim \mathbb{E}[f(\mathcal{R} \cup \{p\}) - f(\mathcal{R} \setminus \{p\})], \quad p \in \mathcal{P}_i. \tag{3.8}$$

Let each agent *propagate* its local variable according to

$$\mathbf{x}_i^-(t+1) = \mathbf{x}_i(t) + \frac{1}{T}\widetilde{\mathbf{v}}_i(t). \tag{3.9}$$

One can make a conceptual connection between (3.9) and the practical implementation of (3.2) discussed earlier. Because the propagation is only based on local information of agent i, next, each agent, by interacting with its neighbors, updates its propagated $\mathbf{x}_i^-(t+1)$ by element-wise maximum seeking

$$\mathbf{x}_i(t+1) = \max_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{x}_j^-(t+1). \tag{3.10}$$

Lemma 3.2.2 below shows that, as one expects, $\mathbf{x}_{ii}(t+1) = \mathbf{x}_{ii}^-(t+1)$, i.e., the corrected component of $\mathbf{x}_i$ corresponding to agent i itself is the propagated value maintained at agent i, and not the estimated value of any of its neighbors.

Lemma 3.2.2. *Assume that the agents follow the distributed Algorithm 4. Let* $\bar{\mathbf{x}}(t) = \max_{i \in \mathcal{A}} \mathbf{x}_i(t)$ *where* $\mathbf{x}_i$ *is given in* (3.4)*. Moreover, Then,* $\bar{\mathbf{x}}(t) = [\mathbf{x}_{11}^\top(t), \cdots, \mathbf{x}_{NN}^\top(t)]^\top$ *at any time step* t*.*

Proof. Since f is a monotone increasing and submodular set function, we have $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\}) \geq 0$ and hence $\widetilde{\nabla F}(\mathbf{x}_i(t))$ has positive entries $\forall i \in \mathcal{A}$. This results in the optimization (3.5) subject to vector space (3.6) to output vector $\widetilde{\mathbf{v}}_i(t) \in P_i(\mathcal{M})$ that has entries greater or equal to zero. Hence, according to the propagation and update rule (3.9) and (3.10), we can conclude that $\mathbf{x}_{ii}(t)$ has increasing elements and only agent i can update it and other agents only copy this value as $\hat{\mathbf{x}}_{ji}(t)$. Therefor, we can conclude that $\hat{x}_{jip}(t) \leq x_{iip}(t)$ for all $p \in \mathcal{P}_i$ which concludes the proof. □

We note that it follows from Lemma 3.2.2 that

$$\mathbf{x}_{jj}(t) = \mathbf{x}_{jj}^{-}(t), \quad j \in \mathcal{A}. \tag{3.11}$$

In the distributed Algorithm 4 at each time step t each agent has its own belief on the probabilities of the policies that is not necessarily the same as the belief of the other agents. The following result establishes the difference between the belief of the agents.

Proposition 3.2.1. *Let agents follow the distributed Algorithm 4. Then, the vectorized membership probability $\mathbf{x}_i(t)$ for each agent $i \in \mathcal{A}$ realized from $\mathcal{F}_i(t)$ satisfy*

$$0 \leq \frac{1}{N}\mathbf{1}.(\bar{\mathbf{x}}(t) - \mathbf{x}_i(t)) \leq \frac{1}{T}d(\mathcal{G}), \tag{3.12a}$$

$$\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t) = \frac{1}{T}\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t), \tag{3.12b}$$

$$\frac{1}{N}\mathbf{1}.(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) = \frac{1}{T}. \tag{3.12c}$$

Proof. f is a monotone increasing and submodular set function therefor $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\}) \geq 0$ and hence $\widetilde{\nabla} F(\mathbf{x}_i(t))$ has positive entries $\forall i \in \mathcal{A}$. Then, because $\widetilde{\mathbf{v}}_j(t) \in P_j(\mathcal{M})$, it follows from (3.5) that $\widetilde{\mathbf{v}}_j(t)$ has non-negative entries, $\widetilde{v}_{jp}(t) \geq 0$, which satisfy $\sum_{p\in\mathcal{P}_j} \widetilde{v}_{jp}(t) = 1$. Therefore, it follows from (3.9) and Lemma 3.2.2 that

$$\mathbf{1}.\mathbf{x}_{jj}(t+1) = \mathbf{1}.\mathbf{x}_{jj}(t) + \frac{1}{T}, \quad j \in \mathcal{A}. \tag{3.13}$$

Using (3.13), we can also write

$$\mathbf{1}.\mathbf{x}_{jj}(t) = \mathbf{1}.\mathbf{x}_{jj}(t - d(\mathcal{G})) + \frac{1}{T}d(\mathcal{G}), \quad j \in \mathcal{A}. \tag{3.14}$$

Furthermore, it follows from Lemma (3.2.2) that for all $\forall p \in \mathcal{P}_j$ and any $i \in \mathcal{A}\backslash\{j\}$ we can

write

$$x_{jjp}(t) \geq \hat{x}_{ijp}(t). \tag{3.15}$$

Also, since every agent $i \in \mathcal{A}\backslash\{j\}$ can be reached from agent $j \in \mathcal{A}$ at most in $d(\mathcal{G})$ hops, it follows from the propagation and update laws (3.9) and (3.10), for all $\forall p \in \mathcal{P}_j$, for any $i \in \mathcal{A}\backslash\{j\}$ that

$$\hat{x}_{ijp}(t) \geq x_{jjp}(t - d(\mathcal{G})). \tag{3.16}$$

Thus, for any $j \in \mathcal{A}$ and $i \in \mathcal{A}\backslash\{j\}$, (3.15) and (3.16) result in

$$\mathbf{1}.\mathbf{x}_{jj}(t) \geq \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \geq \mathbf{1}.\mathbf{x}_{jj}(t - d(\mathcal{G})). \tag{3.17}$$

Next, we can use (3.14) and (3.17) to write

$$\mathbf{1}.\mathbf{x}_{jj}(t) \geq \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \geq \mathbf{1}.\mathbf{x}_{jj}(t) - \frac{1}{T}d(\mathcal{G}), \tag{3.18}$$

for $j \in \mathcal{A}$ and $i \in \mathcal{A}\backslash\{j\}$. Using (3.18) for any $i \in \mathcal{A}$ we can write

$$\sum_{j\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{jj}(t) \geq \mathbf{x}_{ii}(t) + \sum_{j\in\mathcal{A}\backslash\{i\}} \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \geq \sum_{j\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{jj}(t) - \frac{1}{T}Nd(\mathcal{G}). \tag{3.19}$$

Then, using Lemma 3.2.2, from (3.19) we can write

$$\mathbf{1}.\bar{\mathbf{x}}(t) \geq \mathbf{1}.\mathbf{x}_i(t) \geq \mathbf{1}.\bar{\mathbf{x}}(t) - \frac{1}{T}Nd(\mathcal{G}),$$

which ascertains (3.12a). Next, note that from Lemma 3.2.2, we have $\mathbf{x}_{jj}(t) = \mathbf{x}^{-}_{jj}(t)$ for any $j \in \mathcal{A}$. Then, using (3.9) and invoking Lemma 3.2.2, we obtain (3.12b),which, given (3.13), also ascertains (3.12c). □

Because f is normal and monotone increasing, we have the guarantees that $w_p^i \geq 0$. Therefore, without loss of generality, we know that one realization of $\widetilde{\mathbf{v}}_i(t)$ in (3.5) corresponds to $\mathbf{1}_{\{p^\star\}}$ where

$$p^\star = \arg\max_{p \in \mathcal{P}_i} w_p^i. \tag{3.20}$$

Next, let each agent $i \in \mathcal{A}$ propagate its information set according to

$$\mathcal{F}_i^-(t+1) = \mathcal{F}_i(t) \oplus \left\{ (p^\star, \frac{1}{T}) \right\}, \tag{3.21}$$

and update it using a local interaction with its neighbors according to

$$\mathcal{F}_i(t+1) = \underset{j \in \mathcal{N}_i \cup \{i\}}{\mathrm{MAX}} \mathcal{F}_j^-(t+1). \tag{3.22}$$

By definition to $\oplus$ and MAX operators, we have the guarantees that if (p, α_1), then there exists no $\alpha_2 \neq \alpha_1$ that $(p, \alpha_2) \in \mathcal{F}_i$.

Lemma 3.2.3. *For* $\widetilde{\mathbf{v}}_i(t) = \mathbf{1}_{\{p^\star\}}$, $\mathbf{x}_i^-(t+1)$ *and* $\mathbf{x}_i(t+1)$ *computed from, respectively,* (3.9) *and* (3.10) *are the same as* $\mathbf{x}_i^-(t+1)$ *and* $\mathbf{x}_i(t+1)$ *constructed from, respectively,* $\mathcal{F}_i^-(t+1)$ *and* $\mathcal{F}_i(t+1)$ *using* (3.4).

Proof. The proof follows trivially from the definition of the operator $\oplus$ and (3.4). □

Initialized by $\mathcal{F}_i(0) = \emptyset$, $i \in \mathcal{A}$, (3.20), (3.21), and (3.21) where w_p^i is computed via (3.8) constitute a distributed iterative process, formally stated by Algorithm 4, that runs for T steps. At the end of these T steps, as stated in Algorithm 4, each agent $i \in \mathcal{A}$, obtains its suboptimal policy $\bar{\mathcal{P}}_i$ by sampling one policy $\bar{p} \in \mathcal{P}_i$ with the probability given by $\mathbf{x}_{ii}(T)$,

where for $p \in \mathcal{P}_i$,

$$x_{ip}(T) = \begin{cases} \alpha, & (p, \alpha) \in \mathcal{F}_i(T), \\ 0 & \text{otherwise.} \end{cases}$$

The following result gives the convergence guarantee and suboptimality gap of Algorithm 4.

Theorem 3.2.2 (Convergence guarantee and suboptimality gap of Algorithm 4)**.** *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ be normalized, monotone increasing and submodular set function. Let $\mathcal{S}^\star$ to be the optimizer of problem* (3.1)*. Then, the admissible policy set $\bar{\mathcal{P}}$, the output of distributed Algorithm 4, satisfies*

$$\left(1-\frac{1}{\mathrm{e}}\right)\left(1-\left(2N^2 d(\mathcal{G})+\frac{1}{2}N^2+N\right)\frac{1}{T}\right) f(\mathcal{S}^\star) \leq \mathbb{E}[f(\bar{\mathcal{P}})],$$

with the probability of $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$.

Proof. Knowing that $\left|\frac{\partial^2 F}{\partial x_p \partial x_q}\right| \leq f(\mathcal{S}^\star)$ from Lemma 1.5.4 and (3.12c), it follows from Lemma 1.5.7 that

$$F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \nabla F(\bar{\mathbf{x}}(t)).(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) - \frac{1}{2}N^2\frac{1}{T^2}f(\mathcal{S}^\star),$$

which, given (3.12b), leads to

$$F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \frac{1}{T}\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - \frac{1}{2}N^2\frac{1}{T^2}f(\mathcal{S}^\star). \tag{3.23}$$

Next, we note that by definition, $\bar{\mathbf{x}}(t) \geq \mathbf{x}_i(t)$ for any $\forall i \in \mathcal{A}$. Therefore, given (3.12a), by invoking Lemma 1.5.7, for any $i \in \mathcal{A}$ we can write

$$\left|\frac{\partial F}{\partial x_p}(\bar{\mathbf{x}}(t)) - \frac{\partial F}{\partial x_p}(\mathbf{x}_i(t))\right| \leq N\frac{1}{T}d(\mathcal{G}))f(\mathcal{S}^\star), \tag{3.24}$$

for $p \in \{1, \cdots, n\}$. Recall that at each time step t, the realization of $\widetilde{\mathbf{v}}_i(t)$ in (3.5) that Algorithm 4 uses is

$$\widetilde{\mathbf{v}}_i(t) = \mathbf{1}_{p^\star}, \qquad p^\star \in \mathcal{P}_i \text{ is given by (3.20)} \tag{3.25}$$

for every $i \in \mathcal{A}$. Thus, $\mathbf{1}.\widetilde{\mathbf{v}}_i(t) = 1$, $i \in \mathcal{A}$. Consequently, using (3.24) we can write

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - N^2\frac{1}{T}d(\mathcal{G}))f(\mathcal{S}^\star). \tag{3.26}$$

Next, we let

$$\bar{\mathbf{v}}_i(t) = \underset{\mathbf{v}\in P_i(\mathcal{M})}{\text{argmax}}\mathbf{v}.\nabla F(\bar{\mathbf{x}}(t))$$

and

$$\hat{\mathbf{v}}_i(t) = \underset{\mathbf{v}\in P_i(\mathcal{M})}{\text{argmax}}\mathbf{v}.\nabla F(\mathbf{x}_i(t)).$$

Because f is monotone increasing, by virtue of Lemma 1.5.5, $\frac{\partial F}{\partial x_p} \geq 0$, and as such $\bar{\mathbf{v}}_i(t) = \mathbf{1}_{\bar{p}}$ and $\hat{\mathbf{v}}_i(t) = \mathbf{1}_{\hat{p}}$, where $\bar{p} = \arg\max_{p\in\mathcal{P}_i} \frac{\partial F(\bar{\mathbf{x}}(t))}{\partial x_p}$ and $\hat{p} = \arg\max_{p\in\mathcal{P}_i} \frac{\partial F(\mathbf{x}_i(t))}{\partial x_p}$. Therefore, using

$$\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))$$

and

$$\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)),$$

$i \in \mathcal{A}$, and (3.24) we can also write

$$\sum_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - N^2\frac{1}{T}d(\mathcal{G})f(\mathcal{S}^\star), \tag{3.27a}$$

$$\sum_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)). \tag{3.27b}$$

On the other hand, by virtue of Lemma 1.5.9, $\frac{\widetilde{\partial F}}{\partial x_p}(\mathbf{x}_j(t))$, $p \in \mathcal{P}_j$ that each agent $j \in \mathcal{A}$ uses to solve optimization problem (3.20) (equivalently (3.5)) satisfies

$$\left| \frac{\widetilde{\partial F}}{\partial x_p}(\mathbf{x}_j(t)) - \frac{\partial F}{\partial x_p}(\mathbf{x}_j(t)) \right| \leq \frac{1}{2T} f(\mathcal{S}^\star) \tag{3.28}$$

with the probability of $1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_j}$. Using (3.27b) and (3.28), and also that the samples are drawn independently

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) - N\frac{1}{2T} f(\mathcal{S}^\star), \tag{3.29a}$$

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - N\frac{1}{2T} f(\mathcal{S}^\star), \tag{3.29b}$$

with the probability of $\prod_{i\in\mathcal{A}}(1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}$.

From (3.26), (3.27a),(3.29a), and (3.29b) now we can write

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - (2Nd(\mathcal{G})) + 1)N\frac{1}{T} f(\mathcal{S}^\star), \tag{3.30}$$

with the probability of $1 - 2\sum_{i\in\mathcal{A}} \mathrm{e}^{-\frac{1}{8T^2}K_i}$.

Next, let $\mathbf{v}_i^\star$ be the projection of $\mathbf{1}_{\mathcal{S}^\star}$ into $P_i(\mathcal{M})$. Knowing that $P_i(\mathcal{M})$s are disjoint subspaces of $P(\mathcal{M})$ covering the whole space then we can write

$$\mathbf{1}_{\mathcal{S}^\star} = \sum_{i\in\mathcal{A}} \mathbf{v}_i^\star. \tag{3.31}$$

Then, using (3.30), (3.31), and invoking Lemma 3.2.1 and the fact that $\bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq$

$\mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t))$ we obtain

$$\sum_{i\in\mathcal{A}}\widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum_{i\in\mathcal{A}}\mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t)) - (2Nd(\mathcal{G}))+1)N\frac{1}{T}f(\mathcal{S}^\star) =$$
$$\mathbf{1}_{\mathcal{S}^\star}.\nabla F(\bar{\mathbf{x}}(t)) - (2Nd(\mathcal{G}))+1)N\frac{1}{T}f(\mathcal{S}^\star) \geq f(\mathcal{S}^\star) - F(\bar{\mathbf{x}}(t)) - (2Nd(\mathcal{G}))+1)\frac{N}{T}f(\mathcal{S}^\star), \tag{3.32}$$

with the probability of $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}$. Hence, using (3.23) and (3.32), we conclude that

$$F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \frac{1}{T}(f(\mathcal{S}^\star) - F(\bar{\mathbf{x}}(t)) - (2Nd(\mathcal{G})) + \frac{1}{2}N+1)\frac{N}{T^2}f(\mathcal{S}^\star), \tag{3.33}$$

with the probability of $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}$.

Next, let $g(t) = f(\mathcal{S}^\star) - F(\bar{\mathbf{x}}(t))$ and $\beta = (2Nd(\mathcal{G})) + \frac{1}{2}N+1)\frac{N}{T^2}f(\mathcal{S}^\star)$, to rewrite (3.33) as

$$(f(\mathcal{S}^\star) - F(\bar{\mathbf{x}}(t))) - (f(\mathcal{S}^\star) - F(\bar{\mathbf{x}}(t+1))) =$$
$$g(t) - g(t+1) \geq \frac{1}{T}(f(\mathcal{S}^\star) - F(\bar{\mathbf{x}}(t))) - \beta = \frac{1}{T}g(t) - \beta. \tag{3.34}$$

Then from inequality (3.34) we get

$$g(t+1) \leq (1-\frac{1}{T})g(t) + \beta \tag{3.35}$$

with the probability of $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}$. Solving for inequality (3.35) at time T yields

$$g(T) \leq (1-\frac{1}{T})^T g(0) + \beta\sum_{k=0}^{T-1}(1-\frac{1}{T})^k = (1-\frac{1}{T})^T g(0) + T\beta(1-(1-\frac{1}{T})^T) \tag{3.36}$$

with the probability of $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$. Substituting back $g(T) = f(\mathcal{S}^\star) -$

$F(\bar{\mathbf{x}}(T))$ and $g(0) = f(\mathcal{S}^\star) - F(\mathbf{x}(0)) = f(\mathcal{S}^\star)$, in (3.36) we then obtain

$$(1-(1-\frac{1}{T})^T)(f(\mathcal{S}^\star)-T\beta) = (1-(1-\frac{1}{T})^T)(1-(2Nd(\mathcal{G}))+\frac{1}{2}N+1)\frac{N}{T})f(\mathcal{S}^\star)$$
$$\leq F(\bar{\mathbf{x}}(T)), \tag{3.37}$$

with the probability of $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$. By applying $\frac{1}{\mathrm{e}} \geq (1-(1-\frac{1}{T})^T)$, we get

$$(1-\frac{1}{\mathrm{e}})(1-(2Nd(\mathcal{G})+\frac{1}{2}N+1)\frac{N}{T})f(\mathcal{S}^\star) \leq F(\bar{\mathbf{x}}(T)), \tag{3.38}$$

with the probability of $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$.

Given (3.25), from the propagation and update rules (3.9) and (3.10) and Lemma 3.2.2 we can conclude that $\mathbf{1}.\mathbf{x}_{ii}(T) = 1$. Furthermore by defining $\mathcal{R}_{\mathbf{x}_{ii}(T)}$ to be a random set where each member if sampled according to $\mathbf{x}_{ii}(T)$ and from $\mathcal{P}_i$. Since $\mathbf{1}.\mathbf{x}_{ii}(T) = 1$, we can also define $\mathcal{T}_{\mathbf{x}_{ii}(T)}$ to be a random set where only one policy is sampled from $\mathcal{P}_i$ according to $\mathbf{x}_{ii}(T)$, then using Lemma 4.2.4, we can write

$$F(\bar{\mathbf{x}}(T)) = \mathbb{E}[f(\mathcal{R}_{\mathbf{x}(T)})] = \mathbb{E}[f(\mathcal{R}_{\mathbf{x}_{11}(T)} \cup \cdots \cup \mathcal{R}_{\mathbf{x}_{NN}(T)})]$$
$$\leq \mathbb{E}[f(\mathcal{T}_{\mathbf{x}_{11}(T)} \cup \mathcal{R}_{\mathbf{x}_{22}(T)} \cup \cdots \cup \mathcal{R}_{\mathbf{x}_{NN}(T)})]$$
$$\leq \cdots$$
$$\leq \mathbb{E}[f(\mathcal{T}_{\mathbf{x}_{11}(T)} \cup \cdots \cup \mathcal{T}_{\mathbf{x}_{NN}(T)})]$$

which concludes the proof. □

By simplifying the probability statement and dropping the higher order terms, the optimality gap grantee of Theorem 3.2.2 holds with the probability of at least $1 - 2T\,n\,\mathrm{e}^{-\frac{1}{8T^2}\underline{K}}$, $\underline{K} = \min\{K_1, \cdots, K_N\}$; note that $1-2T\,n\,\mathrm{e}^{-\frac{1}{8T^2}\underline{K}} \leq \left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$. Then, it is clear that the probability improves as T and the $\underline{K}$, the number of the samples collected by agents,

increase.

Remark 3.2.2 (Extra communication for improved optimality gap)**.** *Replacing the update step* (3.10) *with* $\mathbf{x}_i(t+1) = \mathbf{y}_i(d(\mathcal{G}))$ *where* $\mathbf{y}_i(0) = \mathbf{x}_i^-(t+1)$ *and*

$$\mathbf{y}_i(m) = \max_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{y}_j(m-1), \quad m \in \{1, \cdots, d(\mathcal{G})\},$$

i.e., starting with $\mathbf{x}_i^-(t+1)$ *and recursively repeating the update step* (3.10) *using the output of the previous recursion for* $d(\mathcal{G})$ *times, each agent* $i \in \mathcal{A}$ *arrives at* $\mathbf{x}_i(t+1) = \bar{\mathbf{x}}(t+1)$ *(recall Lemma 3.2.2). Hence, for this revised implementation, following the proof of Theorem 3.2.2, we observe that* (3.24) *is replaced by* $\left| \frac{\partial F}{\partial x_p}(\bar{\mathbf{x}}(t)) - \frac{\partial F}{\partial x_p}(\mathbf{x}_i(t)) \right| = 0$*, which consequently, leads to*

$$\left(1 - \frac{1}{\mathrm{e}}\right)\left(1 - \left(\frac{1}{2}N^2 + N\right)\frac{1}{T}\right) f(\mathcal{S}^\star) \leq \mathbb{E}[f(\bar{\mathcal{P}})], \tag{3.39}$$

with the probability of $\left(\prod_{i \in \mathcal{A}} (1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$*. This improved optimality gap is achieved by* $(d(\mathcal{G})-1)T$ *extra communication per agent. The optimality bound* (4.59) *is the same bound that is achieved by the centralized algorithm of [4]. To implement this revision, Algorithm 4's step 11 (equivalent to* (3.21)*) should be replaced by* $\mathcal{F}_i = \mathcal{H}_i(d(\mathcal{G}))$*, where* $\mathcal{H}_i(0) = \mathcal{F}_i^-$*, and*

$$\mathcal{H}_i(m) = \underset{j \in \mathcal{N}_i \cup \{i\}}{\mathrm{MAX}} \mathcal{H}_j^-(m-1), \quad m \in \{1, \cdots, d(\mathcal{G})\}. \tag{3.40}$$

3.3 Numerical Example

Consider the multi-agent sensor placement problem introduced in Fig. 3.2 for 5 agents for which $\mathcal{B}_i = \mathcal{B}$, i.e., each agent is able to move to any of the sensor placement points. This sensor placement problem is cast by optimization problem (4.1). The field is a 6 unit by 6 unit

K / T	10000	500	100	50	10	5	1
100	768	768	718	710	718	716	696
20	768	768	718	710	726	716	696
10	661	640	657	640	634	602	551
5	630	630	634	626	583	608	540
1	456	456	456	456	456	456	456

Table 3.1 – The outcome of Algorithm 4 for different iteration and sampling numbers.

square and the feasible sensor locations are the 6 by 6 grid in the center square of the field, see Fig. 3.1. The points of interest are spread around the map (small red dots in Fig. 3.1) in the total number of 900. The sensing zone of the agents $\mathcal{A} = \{a, b, c, d, e\}$ are circles with radii of respectively $\{0.5, 0.6, 0.7, 0.8, 1.5\}$. The agents communicate over a ring graph as shown in Fig. 3.1. We first solve this problem using our proposed distributed Algorithm 4. The results of the simulation for different iteration and sampling numbers are shown in Table 3.1. The algorithm produces good results at a modest number of iteration and sampling numbers (e.g. see $T = 20$ and $K = 500$). Fig. 3.1(g) shows the result of the deployment using Algorithm 4. Next, we solve this algorithm using the sequential greedy algorithm [131] in a decentralized way by first choosing a route $\texttt{SEQ} = \boxed{1} \to \boxed{2} \to \boxed{3} \to \boxed{4} \to \boxed{5}$ that visits all the agents, and then giving `SEQ` to the agents so they follow `SEQ` to share their information in a sequential manner. Figure 3.1(a)-(f) gives 6 possible `SEQ` denoted by the semi-circular arrow inside the networks. The results of running the sequential greedy algorithm over the sequences in Fig. 3.1(a)-(f) is shown in Table 3.2. What stands out about the sequential greedy algorithm

Case	1	2	3	4	5	6
Utility	634	704	699	640	767	760

Table 3.2 – Outcome of sequential greedy algorithm.

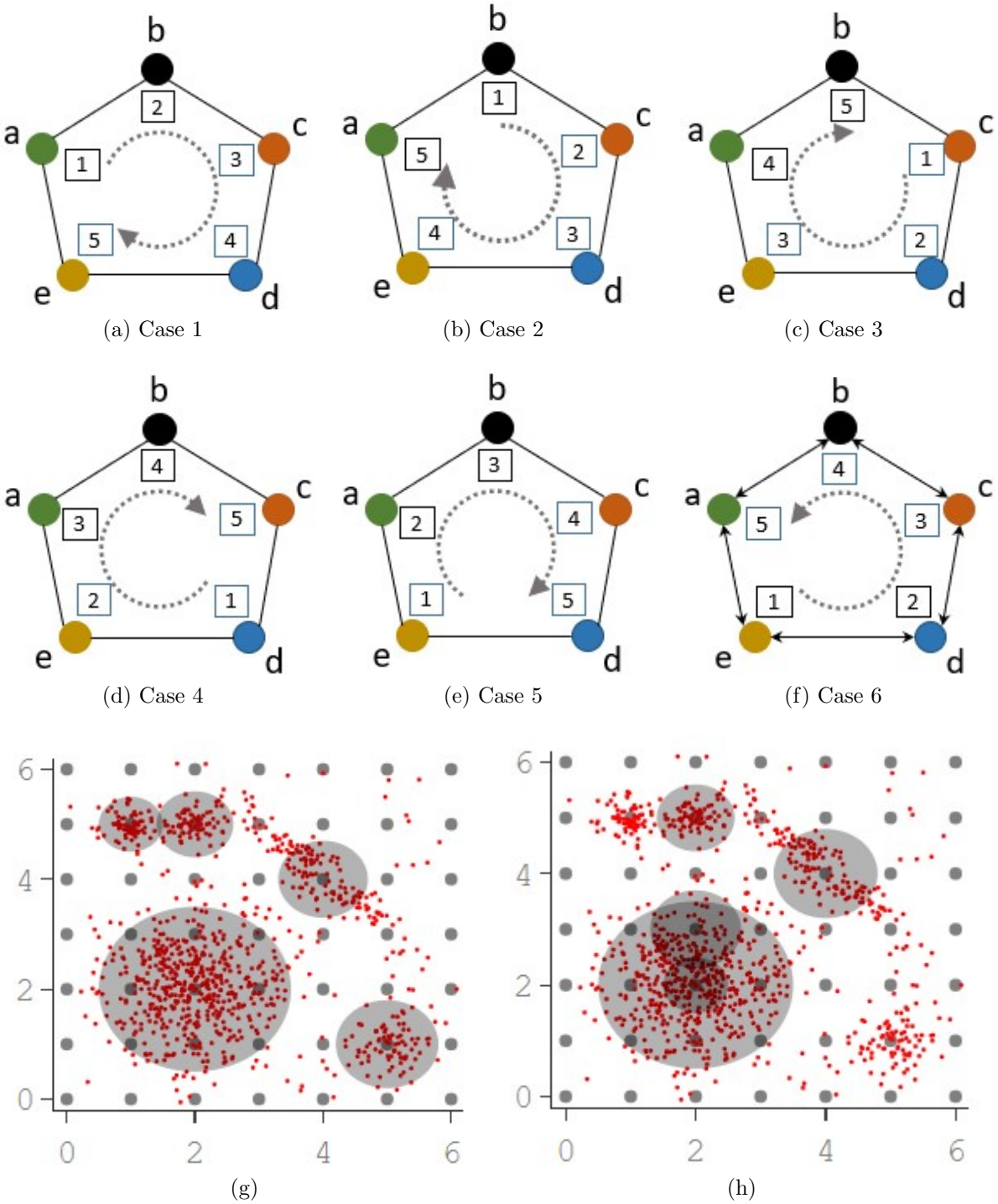
is that the choice of sequence can affect the outcome of the algorithm significantly. We can attribute this inconsistency to the heterogeneity of the sensors' measurement zone. We can see that when sensor e is given the last priority to make its choice, the sequential greedy

algorithm acts poorly. This can be explained by agents with smaller sensing zone picking high-density areas but not being able to cover it fully, see Fig. 3.3(h) which depicts the outcome of a sequential greedy algorithm using the sequence in Case 1. A simpler example justifying this observation is shown in Fig. 3.3 with the two disjoint clusters of points and two sensors. One may suggest to sequence the agents from high to low sensing zone order, however this is not necessarily the best choice as we can see in Table 3.2; the utility of case 6 is less than case 5 (the conjecture of sequencing the agents from strongest to weakest is not valid). Moreover, this ordering may lead to a very long `SEQ` over the communication graph. Interestingly, this inconsistency does not appear in solutions of Algorithm 4 where the agents intrinsically are overcoming the importance of a sequence by deciding the position of the sensors over a time horizon of communication and exchanging their information set.

3.4 Conclusion

We proposed a distributed suboptimal algorithm to solve the problem of maximizing an monotone increasing submodular set function subject to a partitioned matroid constraint. Our problem of interest was motivated by optimal multi-agent sensor placement problems in discrete space. Our algorithm was a practical decentralization of a multilinear extension based algorithm that achieves $(1 - 1/\mathrm{e} - O(1/T))$ optimally gap, which is an improvement over $1/2$ optimality gap that the well-known sequential greedy algorithm achieves. In our numerical example, we compared the outcome obtained by our proposed algorithm with that of a decentralized sequential greedy algorithm which is constructed from assigning a priority sequence to the agents. We showed that the outcome of the sequential greedy algorithm is inconsistent and depends on the sequence. However, our algorithm's outcome due to its iterative nature intrinsically tended to be consistent, which in some ways also explains its better optimally gap over the sequential greedy algorithm. Our future work is to study the

robustness of our proposed algorithm to message dropout.



(a) Case 1 (b) Case 2 (c) Case 3

(d) Case 4 (e) Case 5 (f) Case 6

(g) (h)

Figure 3.1 – Plots (a)-(f) show 6 different `SEQ`s used in the sequential greedy algorithm. Plot (g) shows the outcome of using Algorithm 4 whereas plot (h) shows the outcome of the sequential greedy algorithm when `SEQ` in Case 1 (plot (a)) is used.

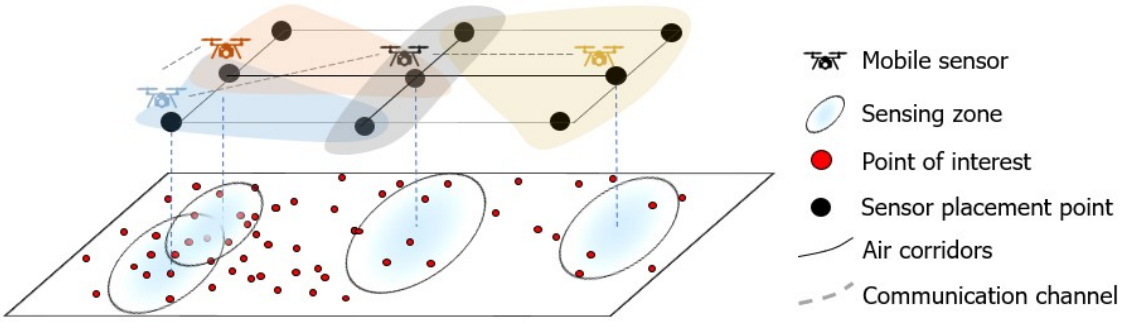


Figure 3.2 – Let the policy set of each mobile sensor $i \in \mathcal{A}$ be $\mathcal{P}_i = \{(i,p)|p \in \mathcal{B}_i\}$, where $\mathcal{B}_i \subset \mathcal{B}$ is the set of the allowable sensor placement points for agent $i \in \mathcal{A}$ out of all the sensor placement points $\mathcal{B}$. Note that by definition, for any two agent $i, j \in \mathcal{A}$, $\mathcal{P}_i \cap \mathcal{P}_j = \emptyset$. The sensors are heterogeneous, in the sense that the size of their sensing zone is different. The objective is to place the sensors in points in $\mathcal{B}$ such that the total number of the observed points of interest is maximized. The utility function, the sum of observed points, is known to be a monotone and increasing submodular function of the agent's sensing zone [1]. This sensor placement problem can be formalized as the optimization problem (4.1). The agents are communicating over a connected undirected graph and their objective is to obtain their respective placement points by interacting only with their communicating neighbors.

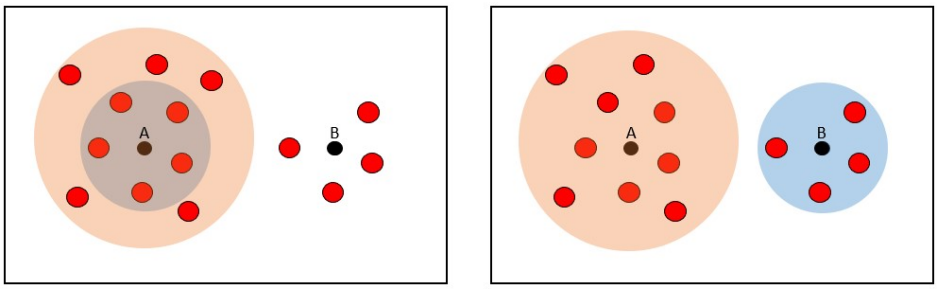


Figure 3.3 – or The sequential greedy algorithm, when the blue agent chooses first assigns both the blue and the orange agents to point A resulting in inferior performance compared to the case that the orange agent chooses first. In the later case, orange agent gets A and the blue agent gets B, which is indeed the optimal solution.

Chapter 4

Distributed Strategy Selection II

Joint utility-maximization problems for multi-agent systems often should be addressed by distributed strategy-selection formulation. Constrained by discrete feasible strategy sets, these problems are broadly formulated as NP-hard combinatorial optimization problems. However, in many cases, it is possible to reformulate this class of problems as constrained submodular set function maximization problems which also belong to the NP-hard domain of problems. A prominent example is the problem of multi-agent mobile sensor dispatching over a discrete domain. This thesis considers a class of submodular optimization problems that consist of maximization of a monotone and submodular set function subject to a partition matroid constraint over a group of networked agents that communicate over a connected undirected graph. We work with the *value oracle* model. Consequently, the only access of the agents to the utility function is through a black box that returns the utility function value given a specific strategy set. We propose a distributed suboptimal polynomial-time algorithm that enables each agent to obtain its respective strategy via local interactions with its neighboring agents. Our solution is a fully distributed gradient-based algorithm using the submodular set functions' multilinear extension followed by a distributed stochastic Pipage rounding procedure. This algorithm results in a strategy set that when the team utility

function is evaluated at the worst case, the utility function value is in $\frac{1}{c}(1-\mathrm{e}^{-c}-O(1/T))$ of the optimal solution with c to be the curvature of the submodular function. An example demonstrates our results.

4.1 Problem Statement

Consider a group of $\mathcal{A}=\{1,...,N\}$ agents with communication and computation capabilities, interacting over a connected undirected graph $\mathcal{G}(\mathcal{A},\mathcal{E})$ where $\mathcal{E}\subset\mathcal{A}\times\mathcal{A}$ is the edge set. Recall that $\mathcal{G}$ is undirected if and only is $(i,j)\in\mathcal{E}$ means that agents i and j can exchange information. An undirected graph is connected if there is a path from each node to every other node in the graph.

Each agent $i\in\mathcal{A}$ has a distinct discrete strategy set $\mathcal{P}_i$, known only to agent i, and wants to choose at most $\kappa_i\in\mathbb{Z}_{>0}$ strategies from $\mathcal{P}_i$ such that a monotone increasing and submodular utility function $f:2^{\mathcal{P}}\to\mathbb{R}_{\geq 0}$, $\mathcal{P}=\bigcup_{i\in\mathcal{A}}\mathcal{P}_i$, evaluated at all the agents' strategy selection is maximized. In other words, the agents aim to solve in a distributed manner the discrete domain optimization problem

$$\max_{\mathcal{R}\subset\mathcal{P},\mathcal{R}\in\mathcal{I}} f(\mathcal{R}) \tag{4.1a}$$

$$\mathcal{I}=\big\{\mathcal{R}\subset\mathcal{P}\;\big|\;|\mathcal{R}\cap\mathcal{P}_i|\leq\kappa_i,\ \forall i\in\mathcal{A}\big\}. \tag{4.1b}$$

The agents' access to the utility function is through a black box that returns $f(\mathcal{R})$ for any given set $\mathcal{R}\subset\mathcal{P}$ (value oracle model). The constraint set (4.1b) is a partition matroid, which restricts the number of strategy choices of each agent $i\in\mathcal{A}$ to κ_i. In a distributed solution, each agent $i\in\mathcal{A}$ should obtain its respective component $\mathcal{R}_i^\star\subset\mathcal{P}_i$ of $\mathcal{R}^\star=\cup_{j=1}^N\mathcal{R}_j^\star$, the optimal solution of (4.1), by interacting only with the agents that are in its communication range.

In the remainder of this chapter, without loss of generality, we assume that global strategy set is given by $\mathcal{P}=\bigcup_{i\in\mathcal{A}}\mathcal{P}_i=\{1,\cdots,n\}$. Also, we assume that the agents' local strategies each are non-empty consecutive integers and ordered such that if $\mathcal{P}_i=\{p,p+1,\cdots,q\}\subset\mathcal{P}$, then $p-1\in\mathcal{P}_{i-1}$ and $q+1\in\mathcal{P}_{i+1}$.

The distributed solution we propose for solving (4.1) relies on a multilinear extension relaxation approach and a rounding procedure. In what follows, we introduce the notation and definitions needed for this approach.

4.1.1 Multilinear Relaxation

The utility function f assigns values to all the subsets of $\mathcal{P}=\bigcup_{i\in\mathcal{A}}\mathcal{P}_i=\{1,\cdots,n\}$. Thus, equivalently, we can regard the set value utility function as a function on the Boolean hypercube $\{0,1\}^n$, i.e., $f:\{0,1\}^n\to\mathbb{R}$. For a submodular function $f:2^{\mathcal{P}}\to\mathbb{R}_{\geq 0}$, its *multilinear extension* $F:[0,1]^n\to\mathbb{R}_{\geq 0}$ in the continuous space is [4]

$$F(\mathbf{x})=\sum_{\mathcal{R}\subset\mathcal{P}}f(\mathcal{R})\prod_{p\in\mathcal{R}}[\mathbf{x}]_p\prod_{p\notin\mathcal{R}}(1-[\mathbf{x}]_p),\quad \mathbf{x}\in[0,1]^n. \tag{4.2}$$

Given $\mathbf{x}\in[0,1]^n$ we can define $\mathcal{R}_{\mathbf{x}}$ to be the random subset of $\mathcal{P}$ in which each element $p\in\mathcal{P}$ is included independently with probability $[\mathbf{x}]_p$ and not included with probability $1-[\mathbf{x}]_p$. Then the multilinear extension F in (4.2) is simply [4]

$$F(\mathbf{x})=\mathbb{E}[f(\mathcal{R}_{\mathbf{x}})], \tag{4.3}$$

where $\mathbb{E}[.]$ indicates the expected value. Taking the derivatives of $F(\mathbf{x})$ yields [4]

$$\frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x})=\mathbb{E}[f(\mathcal{R}_{\mathbf{x}}\cup\{p\})-f(\mathcal{R}_{\mathbf{x}}\setminus\{p\})], \tag{4.4}$$

and

$$\frac{\partial^2 F}{\partial[\mathbf{x}]_p \partial[\mathbf{x}]_q}(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_\mathbf{x} \cup \{p,q\}) - f(\mathcal{R}_\mathbf{x} \cup \{q\} \setminus \{p\}) \\ - f(\mathcal{R}_\mathbf{x} \cup \{p\} \setminus \{q\}) + f(\mathcal{R}_\mathbf{x} \setminus \{p,q\})]. \tag{4.5}$$

Multilinear-extension function $F(\mathbf{x})$, expands the function evaluation of the utility function over the space between the vertices of the Boolean hypercube $0, 1^n$. For a solution via a continuous relaxation method, the effect of partition matroid constraint should also be considered. To do so, the *matroid polytope* for partition matroid is defined as

$$\mathcal{M} = \{\mathbf{x} \in [0,1]^n \,\big|\, \sum\nolimits_{p \in \mathcal{P}_i} [\mathbf{x}]_p \leq \kappa_i, \forall i \in \mathcal{A}\}. \tag{4.6}$$

The matroid polytope $\mathcal{M}$ is the convex hull of the vertices of the hypercube $\{0,1\}^n$ that satisfies the partition matroid constraint (4.1b). Additionally, note that according to (4.2), $F(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{M}$ is a weighted average of values of F at the vertices of the matriod polytope $\mathcal{M}$. Then, equivalently, $F(\mathbf{x})$ at any $\mathbf{x} \in \mathcal{M}$ is a normalized-weighted average of f on the strategies satisfying constraint (4.1b). As such,

$$f(\mathcal{R}^\star) \geq F(\mathbf{x}), \;\; \mathbf{x} \in \mathcal{M},$$

which is equivalent to $f(\mathcal{R}^\star) = \max\limits_{\mathbf{x} \in \mathcal{M}} F(\mathbf{x})$, where $\mathcal{R}^\star$ is the optimizer of problem (4.1) [4]. Therefore, solving the continuous domain optimization problem

$$\max_{\mathbf{x} \in \mathcal{M}} F(\mathbf{x}), \tag{4.7}$$

can lead to finding the $\mathcal{R}^\star$.

A practical implementation of a gradient-based method to solve (4.7) is achieved by using

an Euler discretized implementation with stepsize of $\frac{1}{T}$ in $T \in \mathbb{Z}_{>0}$ steps. A significant challenge in implementing a gradient-based method is the exponential cost of computing the gradient $\nabla F(\mathbf{x})$ whose calculation requires the knowledge of f at each $R \subset 2^{\mathcal{P}}$. The stochastic interpretation (4.3) of the multilinear extension and its derivatives however offer a mechanism to estimate them with a reasonable computational cost via sampling.

4.2 Distributed Submodular Maximization Subject to Partition Matroid

In this section we propose a distributed algorithm for the submodular maximization problem defined in Section 4.1. Our solution relies on the continuous relaxation of the discrete optimization (4.1). We first find a suboptimal solution to the relaxed problem and then propose a rounding method to map this solution to a feasible suboptimal solution for (4.1).

4.2.1 Central Continuous Greedy Algorithm

The constrained gradient ascent

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}) \quad \text{where} \quad \mathbf{v}(\mathbf{x}) = \arg\max_{\mathbf{w}\in\mathcal{M}} (\mathbf{w}.\nabla F(\mathbf{x})), \tag{4.8}$$

initialized at $\mathbf{x}(0) = \mathbf{0}$ to solve the relaxed problem (4.7), can lead to a suboptimal solution for problem (4.1). Since $\mathcal{M}$ is convex and $\mathbf{x}(0) = \mathbf{0} \in \mathcal{M}$, the trajectory $t \mapsto \mathbf{x}$ of (4.8) belongs to $\mathcal{M}$ for $t \in [0, 1]$. The following Lemma provides an essential property of the multilinear extended function F which can be used in quantifying the optimality gap of gradient ascent solver (4.8).

Lemma 4.2.1. *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ be normalized, monotone increasing and submodular set*

function with total curvature (4.21) *of* c. *Given* $\mathcal{P}_1, \mathcal{P}_2 \subset \mathcal{P}$,

$$f(\mathcal{P}_2 \cup \mathcal{P}_1) - f(\mathcal{P}_2) + (1-c)\sum_{p \in \mathcal{P}_1 \cap \mathcal{P}_1} f(\{p\}) \geq (1-c) f(\mathcal{P}_1)$$

holds.

Proof. based on the definition of total curvature (4.21), for $\mathcal{R}, \mathcal{R}' \in \mathcal{P}$ and $p \notin \mathcal{R}, \mathcal{R}'$ we can write

$$c\Delta_f(p|\mathcal{R}') \geq \Delta_f(p|\mathcal{R}') - \Delta_f(p|\mathcal{R})$$

Hence, by taking $\mathcal{P}_1' = \mathcal{P}_1 \setminus \mathcal{P}_2$ and $\mathcal{P}_1' = \{p_{11}, p_{12}, ..., p_{1l}\}$, then we can write

$$c\Delta_f(p_{1i}|\{p_{11}, p_{12}, ..., p_{1(i-1)}\}) \geq$$
$$\Delta_f(p_{1i}|\{p_{11}, p_{12}, ..., p_{1(i-1)}\}) -$$
$$\Delta_f(p_{1i}|\mathcal{P}_1' \cup \{p_{11}, p_{12}, ..., p_{1(i-1)}\}).$$

Therefore we can do a summation over the increments of all members of $\mathcal{P}_1'$

$$\sum_{l}^{i=1} c\Delta_f(p_{1i}|\{p_{11}, p_{12}, ..., p_{1(i-1)}\}) \geq$$
$$\sum_{l}^{i=1} \Delta_f(p_{1i}|\{p_{11}, p_{12}, ..., p_{1(i-1)}\}) -$$
$$\sum_{l}^{i=1} \Delta_f(p_{1i}|\mathcal{P}_2 \cup \{p_{11}, p_{12}, ..., p_{1(i-1)}\}).$$

and by rearranging and knowing that $f(\mathcal{P}_2 \cup \mathcal{P}'_1) = f(\mathcal{P}_2 \cup \mathcal{P}_1)$, we get

$$f(\mathcal{P}_2 \cup \mathcal{P}_1) - f(\mathcal{P}_2) \geq (1-c) f(\mathcal{P}'_1).$$

Then by adding $(1-c)\sum_{p \in \mathcal{P}_1 \cap \mathcal{P}_2} f(p)$ to the both sides of the inequality and knowing that $\mathcal{P}_1 = \mathcal{P}'_1 \cup (\mathcal{P}_1 \cap \mathcal{P}_2)$, we get

$$f(\mathcal{P}_2 \cup \mathcal{P}_1) - f(\mathcal{P}_2) + (1-c) \sum_{p \in \mathcal{P}_1 \cap \mathcal{P}_2} f(p) \geq$$
$$(1-c)(f(\mathcal{P}'_1) + \sum_{p \in \mathcal{P}_1 \cap \mathcal{P}_1} f(p)) \geq (1-c) f(\mathcal{P}_1),$$

which concludes the proof. □

Lemma 4.2.2. *Given that $f : \mathcal{P} \to \mathbb{R}_{\geq 0}$ is an increasing and submodular set function with total curvature of c and multi-linear extension $F : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$. Assuming that $\mathcal{R}^\star$ is the optimizer of the problem* (4.1) *subject to matroid $\mathcal{M}$, then $\forall \mathbf{x} \in P(\mathcal{M})$*

$$\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\mathbf{x}) \geq f(\mathcal{R}^\star) - cF(\mathbf{x}),$$

holds.

Proof. Taking $\mathbf{1}_{\mathcal{R}^\star}$ to be the membership indicator vector of optimizer $\mathcal{R}^\star$, then we can write

$$\begin{aligned}
\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\mathbf{x}) &= \sum_{p \in \mathcal{R}^\star} \mathbb{E}[f(\mathcal{R}_\mathbf{x} \cup \{p\}) - f(\mathcal{R}_\mathbf{x} \backslash \{p\})] \\
&= \sum_{p \in \mathcal{R}^\star} \mathbb{E}[f(\mathcal{R}_\mathbf{x} \cup \{p\}) - f(\mathcal{R}_\mathbf{x}) + f(\mathcal{R}_\mathbf{x}) - f(\mathcal{R}_\mathbf{x} \backslash \{p\})] \\
&= \mathbb{E}[\sum_{p \in \mathcal{R}^\star} \big((f(\mathcal{R}_\mathbf{x} \cup \{p\}) - f(\mathcal{R}_\mathbf{x})) + (f(\mathcal{R}_\mathbf{x}) - f(\mathcal{R}_\mathbf{x} \backslash \{p\}))\big)]. \qquad (4.9)
\end{aligned}$$

Knowing that $f(\mathcal{R}_{\mathbf{x}}) - f(\mathcal{R}_{\mathbf{x}} \backslash \{p\})) = 0$ if $p \notin \mathcal{R}_{\mathbf{x}}$ then (4.9) is reduced in

$$\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\mathbf{x}) = \mathbb{E}[\sum_{p \in \mathcal{R}^\star} \Delta_f(p|\mathcal{R}_{\mathbf{x}}) + \sum_{p \in \mathcal{R}^\star \cap \mathcal{R}_{\mathbf{x}}} \Delta_f(p|\mathcal{R}_{\mathbf{x}} \backslash \{p\})] \tag{4.10}$$

Since f is an increasing and submodular function therefor by assuming $\mathcal{R}^\star = \{s_1^\star, \cdots, s_o^\star\}$ and using (1.1) the following holds.

$$\begin{aligned} f(\mathcal{R}_{\mathbf{x}} \cup \mathcal{R}^\star) &= f(\mathcal{R}_{\mathbf{x}}) + \sum_{k=1}^{o} \Delta_f(s_i|\mathcal{R}_{\mathbf{x}} \cup \{s_1^\star, \cdots, s_{k-1}^\star\}) \\ &\leq f(\mathcal{R}_{\mathbf{x}}) + \sum_{p \in \mathcal{R}^\star} \Delta_f(p|\mathcal{R}_{\mathbf{x}}) \end{aligned} \tag{4.11}$$

Furthermore, by the definition of total curvature (4.21) we have

$$\sum_{p \in \mathcal{R}^\star \cap \mathcal{R}_{\mathbf{x}}} (1-c) f(\{P\}) \leq \sum_{p \in \mathcal{R}^\star \cap \mathcal{R}_{\mathbf{x}}} \Delta_f(p|\mathcal{R}_{\mathbf{x}} \backslash \{p\}). \tag{4.12}$$

By putting (4.10),(4.11) and (4.12) together we get

$$\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\mathbf{x}) \geq \mathbb{E}[f(\mathcal{R}_{\mathbf{x}} \cup \mathcal{R}^\star) - f(\mathcal{R}_{\mathbf{x}}) + \sum_{p \in \mathcal{R}^\star \cap \mathcal{R}_{\mathbf{x}}} (1-c) f(\{p\})]. \tag{4.13}$$

Definition of curvature (4.21) also implies that

$$f(\mathcal{R}_{\mathbf{x}} \cup \mathcal{R}^\star) + \sum_{p \in \mathcal{R}^\star \cap \mathcal{R}_{\mathbf{x}}} (1-c) f(\{p\}) \geq f(\mathcal{R}^\star) + (1-c) f(\mathcal{R}_{\mathbf{x}}) \tag{4.14}$$

Therefor, (4.13) and (4.14) results in

$$\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\mathbf{x}) \geq \mathbb{E}[f(\mathcal{R}^\star) - c f(\mathcal{R}_{\mathbf{x}})] = f(\mathcal{R}^\star) + F(\mathbf{x})$$

which concludes the proof. □

Due to Lemma 4.2.2, ascent direction in (4.8) satisfies

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}).\nabla F(\mathbf{x}) \geq f(\mathcal{R}^\star) - c\, F(\mathbf{x}). \tag{4.15}$$

From (4.15), we conclude that (4.8) results in $F(\mathbf{x}(1)) \geq \frac{1}{c}(1 - \mathrm{e}^{-c})\, f(\mathcal{R}^\star)$.

4.2.2 Distributed Discrete Gradient Ascent Solution

In the distributed setting described in our problem definition, every agent initially has access only to its own strategy set. Let every agent $i \in \mathcal{A}$ maintain and evolve a local copy of the membership probability vector as $\mathbf{x}_i(t) \in \mathbb{R}^n$. Since $\mathcal{P} = \{1, \cdots, n\}$ is sorted agent-wise, we denote $\mathbf{x}_i(t) = [\hat{\mathbf{x}}_{i1}^\top(t), \cdots, \mathbf{x}_{ii}^\top(t), \cdots, \hat{\mathbf{x}}_{iN}^\top(t)]^\top \in \mathbb{R}^n$ where $\mathbf{x}_{ii}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_i|}$ is the membership probability vector of agent i's own strategy with entries of $[\mathbf{x}_i(t)]_p,\ p \in \mathcal{P}_i$ at iteration $t \in \{0, 1, \cdots, T\},\ T \in \mathbb{Z}_{>0}$, while $\hat{\mathbf{x}}_{ij}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_j|}$ is the local estimate of the membership probability vector of agent j by agent i with entries of $[\mathbf{x}_i(t)]_p,\ p \in \mathcal{P}_j,\ j \in \mathcal{A} \setminus \{i\}$. Every agent $i \in \mathcal{A}$ initializes at $\mathbf{x}_i(0) = \mathbf{0}$ and implements the *propagation* and *update* steps

$$\mathbf{x}_i^-(t+1) = \mathbf{x}_i(t) + \frac{1}{T}\widetilde{\mathbf{v}}_i(t), \tag{4.16a}$$

$$\mathbf{x}_i(t+1) = \max_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{x}_j^-(t+1), \tag{4.16b}$$

where

$$\widetilde{\mathbf{v}}_i(t) = \operatorname*{argmax}_{\mathbf{w} \in \mathcal{M}_i} \mathbf{w}.\widetilde{\nabla F}(\mathbf{x}_i(t)) \tag{4.17}$$

with

$$\mathcal{M}_i = \Big\{ \mathbf{w} \in [0,1]^n \Big|\ \mathbf{1}^\top.\mathbf{w} \leq \kappa_i\ , [\mathbf{w}]_p = 0,\ \forall p \in \mathcal{P} \backslash \mathcal{P}_i \Big\}. \tag{4.18}$$

The vector $\widetilde{\nabla F}(\mathbf{x}_i(t))$ is the empirical estimate of $\nabla F(\mathbf{x}_i(t))$. At time step t, each agent $i \in \mathcal{A}$ generates K_i independent samples $\{\mathcal{R}_i^k(t)\}_{k=1}^{K_i}$ of random set $\mathcal{R}_{\mathbf{x}_i(t)}$ drawn according to membership probability vector $\mathbf{x}_i(t)$ from $\mathcal{P}$ and empirically computes gradient vector $\nabla F(\mathbf{x}_i(t)) \in \mathbb{R}^n_{\geq 0}$, which according to (4.4) is defined element-wise as

$$\left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p = \left(\sum_{k=1}^{K_i} f(\mathcal{R}_i^k(t) \cup \{p\}) - f(\mathcal{R}_i^k(t) \setminus \{p\})\right) \Big/ K_i,$$

$p \in \mathcal{P} = \{1, \cdots, n\}$.

In the propagation step (4.16a) agent i takes a step along a feasible gradient ascent direction in its own local polytope (4.18). Because the propagation is only based on the local information of agent i, in the update step (4.16b), the propagated $\mathbf{x}_i^-(t+1)$ of each agent $i \in \mathcal{A}$ is updated by element-wise maximum seeking among its neighbors. Lemma 4.2.3 below shows that, as expected,

$$\mathbf{x}_{ii}(t) = \mathbf{x}_{ii}^-(t), \quad i \in \mathcal{A},$$

i.e., the corrected component of $\mathbf{x}_i$ corresponding to agent i itself is the propagated value maintained at agent i, and not the estimated value of any of its neighbors.

Lemma 4.2.3. *Let the agents propagate and update their local membership probability vector according to* (4.16a) *and* (4.16b). *Let*

$$\bar{\mathbf{x}}(t) = \max_{i \in \mathcal{A}} \mathbf{x}_i(t). \tag{4.19}$$

Then, at any $t \in \{0, 1, \cdots, T\}$, we have

$$\bar{\mathbf{x}}(t) = [\mathbf{x}_{11}^\top(t), \cdots, \mathbf{x}_{NN}^\top(t)]^\top. \tag{4.20}$$

Proof. Since f is monotone increasing and submodular, we have $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus$

$\{p\}) \geq 0$ and hence $\widetilde{\nabla F}(\mathbf{x}_i(t))$ has positive entries $\forall i \in \mathcal{A}$. Thus, $\widetilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$, the optimizer of the optimization (4.17) has nonnegative entries. Hence, according to the propagation and update rule (4.16a) and (4.16b), we can conclude that $\mathbf{x}_{ii}(t)$ has increasing elements and only agent i can update it and other agents only copy this value as $\hat{\mathbf{x}}_{ji}(t)$. Therefore, we can conclude that $[\hat{\mathbf{x}}_{ji}]_p(t) \leq [\mathbf{x}_{ii}]_p(t)$ for all $p \in \mathcal{P}_i$ which concludes the proof. □

We interpret $\bar{\mathbf{x}}(t)$ as the *global probability membership vector* of the network. Next lemma states that both $\mathbf{x}_i(t)$ and $\bar{\mathbf{x}}(t)$ belong to $\mathcal{M}$ for any $t \in \{0, \cdots, T\}$, i.e., the trajectories of the local and global membership probability vectors never leave the matroid polytope.

Lemma 4.2.4. *Let the agents propagate and update their local membership probability vector according to* (4.16a) *and* (4.16b). *Then, (a)* $\mathbf{x}_i(t),\ \bar{\mathbf{x}}(t) \in \mathcal{M}$ *at any* $t \in \{0, 1, \cdots, T\}$*; (b)* $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i$, $\mathbf{1}.\hat{\mathbf{x}}_{ij}(T) \leq \kappa_j$, $j \in \mathcal{A} \setminus \{i\}$, *and* $\mathbf{x}_i(T) \in [0,1]^n$.

Proof. The proof follows from a mathematical induction argument. The base case $\mathbf{x}_i(0) = \mathbf{0} \in \mathcal{M}$ and $\bar{\mathbf{x}}(0) = \mathbf{0} \in \mathcal{M}$ is trivially true. We take it to be true that at time t and for each agent $i \in \mathcal{A}$ it hold that $\mathbf{x}_i(t) \in \mathcal{M}$ with

$$\mathbf{1}.\mathbf{x}_{ii}(t) = \frac{t}{T}\kappa_i, \quad \text{and} \quad \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \leq \frac{t}{T}\kappa_j, \quad j \in \mathcal{A} \setminus \{i\}.$$

for $t < T$ and $\widetilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$ satisfying

$$\sum\nolimits_{p\in\mathcal{P}_i}[\widetilde{\mathbf{v}}_i(t)]_p = \kappa_i, \quad \text{and} \quad \sum\nolimits_{p\in\mathcal{P}_j}[\widetilde{\mathbf{v}}_i(t)]_p = 0,\ j \in \mathcal{A}\setminus\{i\}.$$

Since $[\widetilde{\nabla F}(\mathbf{x}_i(t))]_p \geq 0 \quad p \in \mathcal{P}$, then by propagation rule (4.16a), we establish that

$$\mathbf{1}.\mathbf{x}^-_{ii}(t+1) = \frac{t+1}{T}\kappa_i,$$
$$\mathbf{1}.\hat{\mathbf{x}}^-_{ij}(t+1) \leq \frac{t}{T}\kappa_j, \quad j \in \mathcal{A} \setminus \{i\}.$$

A a result of $\mathcal{M}_i$, $i \in \mathcal{A}$ being disjoint convex subspaces of $\mathcal{M}$, the update rule (4.16b) leads to

$$\begin{aligned}
&\mathbf{1}.\mathbf{x}_{ii}(t+1) = \frac{t+1}{T}\kappa_i, \\
&\mathbf{1}.\hat{\mathbf{x}}_{ij}(t+1) \leq \frac{t+1}{T}\kappa_j, \quad j \in \mathcal{A} \setminus \{i\}.
\end{aligned}$$

Therefore, we conclude that $\mathbf{x}_i(t+1) \in \mathcal{M}$. Moreover, by the definition of $\bar{\mathbf{x}}(t)$ in (4.20) and $\mathcal{M}_i$, $i \in \mathcal{A}$ being disjoint convex subspaces of $\mathcal{M}$, we deduct that

$$\mathbf{1}.\bar{\mathbf{x}}(t) = \sum\nolimits_{i\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{ii}(t+1) = \frac{t+1}{T}\kappa_i \quad i \in \mathcal{A},$$

for $t < T$ and therefore, $\bar{\mathbf{x}}(t+1) \in \mathcal{M}$. We conclude the proof of (a) by induction and trivially (b) follows. □

The following theorem quantifies the optimality gap of $F(\bar{\mathbf{x}}(T))$ with respect to the solution of the main problem (4.1). To characterize this optimality gap we take into account the total curvature of the utility function, defined as

$$c = 1 - \min_{\mathcal{S}\subset\mathcal{P},\, p\notin\mathcal{S}} \frac{\Delta_f(p|\mathcal{S})}{\Delta_f(p|\emptyset)}. \tag{4.21}$$

The total curvature $c \in [0,1]$ of a submodular set function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ shows the worst-case increase in the value of the function when member p is added.

Theorem 4.2.1 (Optimality gap)**.** *Let the agents propagate and update their local membership probability vector according to* (4.16)*. Let* $\kappa = \sum_{i\in\mathcal{A}} \kappa_i$*,* K_i *be number of samples agent* i *used to compute* $\widetilde{\nabla F}(\mathbf{x}_i(t))$*, and* $\mathcal{R}^\star$ *be the optimizer of problem* (4.1)*. Then, with the probability of at least* $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\right)^T$*, the global probability membership vector*

$\bar{\mathbf{x}}(T)$ *satisfies*

$$F(\bar{\mathbf{x}}(T)) \geq \beta f(\mathcal{R}^\star), \tag{4.22}$$

where

$$\beta = \frac{1}{c}(1-\mathrm{e}^{-c})(1-(2\,c\,\kappa\,d(\mathcal{G})+\frac{c\,\kappa}{2}+1)\frac{\kappa}{T}). \tag{4.23}$$

Proof. Knowing that $\left|\frac{\partial^2 F}{\partial[\mathbf{x}]_p \partial[\mathbf{x}]_q}\right| \leq c f(\mathcal{R}^\star)$ from Lemma 1.5.5, and (4.40c), it follows from Lemma 1.5.7 that $F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \nabla F(\bar{\mathbf{x}}(t)).(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) - \frac{\kappa^2}{2T^2} c f(\mathcal{R}^\star)$, which, given (4.40b), leads to

$$\begin{aligned} &F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \\ &\quad \frac{1}{T}\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - \frac{\kappa^2}{2T^2} c f(\mathcal{R}^\star). \end{aligned} \tag{4.24}$$

By definition, $\bar{\mathbf{x}}(t) \geq \mathbf{x}_i(t)$ for any $\forall i \in \mathcal{A}$. Therefore, given (4.40a), by invoking Lemma 1.5.7, for any $i \in \mathcal{A}$ we can write

$$\left|\frac{\partial F}{\partial[\mathbf{x}]_p}(\bar{\mathbf{x}}(t)) - \frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_i(t))\right| \leq \frac{\kappa}{T} d(\mathcal{G}) c f(\mathcal{R}^\star), \tag{4.25}$$

for $p \in \{1, \cdots, n\}$. Recall that at each time step t, the realization of $\widetilde{\mathbf{v}}_i(t)$ in (4.17) that Algorithm 5 uses for $\{p_1^\star, \cdots, p_{\kappa_i}^\star\} \in \mathcal{P}_i$ is

$$\widetilde{\mathbf{v}}_i(t) = \mathbf{1}_{\{p_1^\star, \cdots, p_{\kappa_i}^\star\}}, \tag{4.26}$$

for every $i \in \mathcal{A}$. Thus, $\mathbf{1}.\widetilde{\mathbf{v}}_i(t) = \kappa_i$, $i \in \mathcal{A}$. Consequently, using (4.25) we can write

$$\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - \frac{\kappa^2}{T} d(\mathcal{G}))cf(\mathcal{R}^\star). \tag{4.27}$$

Next, we let $\bar{\mathbf{v}}_i(t) = \operatorname{argmax}_{\mathbf{v}\in\mathcal{M}_i} \mathbf{v}.\nabla F(\bar{\mathbf{x}}(t))$ and $\hat{\mathbf{v}}_i(t) = \operatorname{argmax}_{\mathbf{v}\in\mathcal{M}_i} \mathbf{v}.\nabla F(\mathbf{x}_i(t))$. Then, using $\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))$ and $\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))$, $i \in \mathcal{A}$, and (4.25) we can also write

$$\sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - \frac{\kappa^2}{T} d(\mathcal{G})cf(\mathcal{R}^\star), \tag{4.28a}$$

$$\sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)). \tag{4.28b}$$

On the other hand, by virtue of Lemma 1.5.9, $\left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p$, $p \in \mathcal{P}_i$ that each agent $i \in \mathcal{A}$ uses to solve optimization problem (4.56) (equivalently (4.17)) satisfies

$$\left| \left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_i(t)) \right| \leq \frac{1}{2T} f(\mathcal{R}^\star) \tag{4.29}$$

with the probability of at least $1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_i}$. Using (4.28b) and (4.29), and because the samples are drawn independently, we obtain

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) - \frac{\kappa}{2T} f(\mathcal{R}^\star), \tag{4.30a}$$

$$\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - \frac{\kappa}{2T} f(\mathcal{R}^\star), \tag{4.30b}$$

with the probability of at least $\prod_{i\in\mathcal{A}} (1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$.

From (4.27), (4.28a),(4.30a), and (4.30b) now we can write

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - (2\kappa d(\mathcal{G})) + 1)\frac{\kappa}{T} f(\mathcal{R}^\star), \tag{4.31}$$

with the probability of at least $1 - 2\sum_{i\in\mathcal{A}} \mathrm{e}^{-\frac{1}{8T^2}K_i}$.

Next, let $\mathbf{v}_i^\star$ be the projection of $\mathbf{1}_{\mathcal{R}^\star}$ into $\mathcal{M}_i$. Knowing that $\mathcal{M}_i$'s are disjoint sub-spaces of $\mathcal{M}$ covering the whole space then we can write

$$\mathbf{1}_{\mathcal{R}^\star} = \sum_{i\in\mathcal{A}} \mathbf{v}_i^\star. \tag{4.32}$$

Then, using (4.31), (4.32), and invoking Lemma 4.2.2 and the fact that $\bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t))$ we obtain

$$\begin{aligned}
&\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \\
&\sum_{i\in\mathcal{A}} \mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G}) + 1)\frac{\kappa}{T} f(\mathcal{R}^\star) = \\
&\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G}) + 1)\frac{\kappa}{T} f(\mathcal{R}^\star) \geq \\
&f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G}) + 1)\frac{\kappa}{T} f(\mathcal{R}^\star),
\end{aligned} \tag{4.33}$$

with the probability of at least $\prod_{i\in\mathcal{A}}(1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$. Hence, using (4.24) and (4.33), we conclude that

$$\begin{aligned}
F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \frac{1}{T}(f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t)) - \\
(2c\kappa d(\mathcal{G})) + \frac{c\kappa}{2} + 1)\frac{\kappa}{T^2} f(\mathcal{R}^\star),
\end{aligned} \tag{4.34}$$

with the probability of at least $\prod_{i\in\mathcal{A}}(1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$. Next, let $g(t) = f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t))$

and $\beta = (2c\kappa d(\mathcal{G})) + \frac{c\kappa}{2} + 1)\frac{\kappa}{T^2} f(\mathcal{R}^\star)$, to rewrite (4.34) as

$$
\begin{aligned}
&(f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t))) - (f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t+1))) = \\
&\quad g(t) - g(t+1) \geq \frac{c}{T}(f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t))) - c\beta = \\
&\quad \frac{c}{T} g(t) - c\beta. \qquad (4.35)
\end{aligned}
$$

Then from inequality (4.35) we get

$$g(t+1) \leq (1 - \frac{c}{T}) g(t) + c\beta \qquad (4.36)$$

with the probability of at least $\prod_{i \in \mathcal{A}} (1 - 2\mathrm{e}^{-\frac{1}{8T^2} K_i})^{|\mathcal{P}_i|}$. Solving for inequality (4.36) at time T yields

$$
\begin{aligned}
g(T) \leq & (1 - \frac{c}{T})^T g(0) + \beta \sum\nolimits_{k=0}^{T-1} (1 - \frac{c}{T})^k \\
= & (1 - \frac{c}{T})^T g(0) + T\beta(1 - (1 - \frac{c}{T})^T) \qquad (4.37)
\end{aligned}
$$

with the probability of at least $\left(\prod_{i \in \mathcal{A}} (1 - 2\mathrm{e}^{-\frac{1}{8T^2} K_i})^{|\mathcal{P}_i|}\right)^T$. Substituting back $g(T) = f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(T))$ and $g(0) = f(\mathcal{R}^\star) - cF(\mathbf{x}(0)) = f(\mathcal{R}^\star)$, in (4.37) we then obtain

$$
\begin{aligned}
&\frac{1}{c}(1 - (1 - \frac{1}{T})^T)(f(\mathcal{R}^\star) - T\beta) = \\
&\quad \frac{1}{c}(1 - (1 - \frac{1}{T})^T)(1 - (2c\kappa d(\mathcal{G}) + \frac{c\kappa}{2} + 1)\frac{\kappa}{T}) f(\mathcal{R}^\star) \\
&\quad \leq F(\bar{\mathbf{x}}(T)), \qquad (4.38)
\end{aligned}
$$

with the probability of at least $\left(\prod_{i \in \mathcal{A}} (1 - 2\mathrm{e}^{-\frac{1}{8T^2} K_i})^{|\mathcal{P}_i|}\right)^T$. By applying $\mathrm{e}^{-c} \geq (1 - (1 - \frac{c}{T})^T)$, we get

$$\frac{1}{c}(1 - \mathrm{e}^{-c})(1 - (2c\kappa d(\mathcal{G}) + \frac{c\kappa}{2} + 1)\frac{\kappa}{T}) f(\mathcal{R}^\star) \leq F(\bar{\mathbf{x}}(T)), \qquad (4.39)$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\right)^T$. which concludes the proof. $\square$

Notice that since

$$1-2T\,n\,\mathrm{e}^{-\frac{1}{8T^2}\underline{K}} \leq \left(\prod\nolimits_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\right)^T,$$

where $\underline{K} = \min\{K_1, \cdots, K_N\}$, the probability of the bound (4.22) improves as T, and the number of the samples collected by the agents $\underline{K}$ increase.

The following result establishes the difference between each agent i's local copy of the membership probability vector $\mathbf{x}^i(t)$ and the global probability membership vector of the network $\bar{\mathbf{x}}(t)$. It also shows how the global probability membership vector evolves when agents implement (4.16). This result is instrumental in establishing proof of Theorem 6.1.2.

Proposition 4.2.2. *Let the agents propagate and update their local membership probability vector according to* (4.16a) *and* (4.16b). *Then, the membership probability* $\mathbf{x}_i(t)$, $t \in \{0, \cdots, T\}$, *for each agent* $i \in \mathcal{A}$ *satisfies*

$$0 \leq \frac{1}{\kappa}\mathbf{1}.(\bar{\mathbf{x}}(t) - \mathbf{x}_i(t)) \leq \frac{1}{T}d(\mathcal{G}), \tag{4.40a}$$

$$\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t) = \frac{1}{T}\sum\nolimits_{i\in\mathcal{A}}\widetilde{\mathbf{v}}_i(t), \tag{4.40b}$$

$$\frac{1}{\kappa_i}\mathbf{1}.(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) = \frac{1}{T}, \tag{4.40c}$$

where $\kappa = \sum_{i\in\mathcal{A}}\kappa_i$ *and* $\bar{\mathbf{x}}(t)$ *is given by equation* (4.20).

Proof. f is a monotone increasing and submodular set function therefore $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\}) \geq 0$ and hence $\widetilde{\nabla F}(\mathbf{x}_i(t))$ has positive entries $\forall i \in \mathcal{A}$. Then, because $\widetilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$, it follows from (4.17) that $\widetilde{\mathbf{v}}_i(t)$ has non-negative entries, $[\widetilde{\mathbf{v}}_i(t)]_p \geq 0$ which satisfy $\sum_{p\in\mathcal{P}_i}[\widetilde{\mathbf{v}}_i(t)]_p = \kappa_i$. Therefore, it follows from (4.16a) and Lemma 4.2.3 that

$$\mathbf{1}.\mathbf{x}_{ii}(t+1) = \mathbf{1}.\mathbf{x}_{ii}(t) + \frac{\kappa_i}{T}, \quad i \in \mathcal{A}. \tag{4.41}$$

Using (4.41) recursively for $d(\mathcal{G})$ steps, we can also write

$$\mathbf{1}.\mathbf{x}_{ii}(t) = \mathbf{1}.\mathbf{x}_{ii}(t - d(\mathcal{G})) + \frac{\kappa_i}{T} d(\mathcal{G}), \quad i \in \mathcal{A}. \tag{4.42}$$

Furthermore, it follows from Lemma (4.2.3) that for all $\forall p \in \mathcal{P}_i$ and any $j \in \mathcal{A}\backslash\{i\}$, we can write

$$[\mathbf{x}_i(t)]_p \geq [\mathbf{x}_j(t)]_p. \tag{4.43}$$

Also, since every agent $j \in \mathcal{A}\backslash\{i\}$ can be reached from agent $i \in \mathcal{A}$ at most in $d(\mathcal{G})$ hops, it follows from the propagation and update laws (4.16a) and (4.16b), for all $\forall p \in \mathcal{P}_i$, for any $j \in \mathcal{A}\backslash\{i\}$ that

$$[\mathbf{x}_j(t)]_p \geq [\mathbf{x}_i(t - d(\mathcal{G}))]_p(t - d(\mathcal{G})). \tag{4.44}$$

Thus, for $i \in \mathcal{A}$ and $j \in \mathcal{A}\backslash\{i\}$, (4.43) and (4.44) result in

$$\mathbf{1}.\mathbf{x}_{ii}(t) \geq \mathbf{1}.\hat{\mathbf{x}}_{ji}(t) \geq \mathbf{1}.\mathbf{x}_{ii}(t - d(\mathcal{G})). \tag{4.45}$$

Next, we can use (4.42) and (4.45) to write

$$\mathbf{1}.\mathbf{x}_{ii}(t) \geq \mathbf{1}.\hat{\mathbf{x}}_{ji}(t) \geq \mathbf{1}.\mathbf{x}_{ii}(t) - \frac{\kappa_i}{T} d(\mathcal{G}), \tag{4.46}$$

for $i \in \mathcal{A}$ and $j \in \mathcal{A}\backslash\{i\}$. Using (4.46) for any $i \in \mathcal{A}$ we can write

$$\begin{aligned} &\textstyle\sum_{l\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{ll}(t) \geq \mathbf{1}.\mathbf{x}_{ii}(t) + \sum_{j\in\mathcal{A}\backslash\{i\}} \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \geq \\ &\textstyle\sum_{l\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{ll}(t) - \frac{\kappa_l}{T} d(\mathcal{G}). \end{aligned} \tag{4.47}$$

Then, using Lemma 4.2.3, from (4.47) we can write

$$\mathbf{1}.\bar{\mathbf{x}}(t) \geq \mathbf{1}.\mathbf{x}_i(t) \geq \mathbf{1}.\bar{\mathbf{x}}(t) - \frac{\kappa}{T} d(\mathcal{G}),$$

with $\kappa = \sum_{i \in \mathcal{A}} \kappa_i$, which ascertains (4.40a). Next, note that from Lemma 4.2.3, we have $\mathbf{x}_{jj}(t) = \mathbf{x}_{ii}^{-}(t)$ for any $i \in \mathcal{A}$. Then, using (4.16a) and invoking Lemma 4.2.3, we obtain (4.40b), which, given (4.41), also ascertains (4.40c). □

Distributed Pipage Rounding Procedure

The final output of a distributed solver for problem (4.1) must be a set $\bar{\mathcal{R}}$ that belongs to $\mathcal{I}$ defined in (4.1b). Recall that strategies corresponding to the vertices of the matroid polytope $\mathcal{M}$ correspond to admissible strategy set $\mathcal{I}$. However, $\bar{\mathbf{x}}(T)$ is a fractional point in $\mathcal{M}$. Moreover, only part of $\bar{\mathbf{x}}(T)$ is available at each agent $i \in \mathcal{A}$. In what follows, we propose a distributed rounding procedure that without any communication among the agents, enables each agent $i \in \mathcal{A}$ to round its $\mathbf{x}_{ii}(T)$, and use this rounded probability membership vector to make its local strategy choice $\bar{\mathcal{R}}_i$ such that $\cup_{i \in \mathcal{A}} \bar{\mathcal{R}}_i = \bar{\mathcal{R}} \in \mathcal{I}$.

Let each agent $i \in \mathcal{A}$ initialize its local rounded membership vector $\mathbf{y}_{ii} \in \mathbb{R}^{|\mathcal{P}_i|}$ at $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$. Then, by virtue of Lemma 4.2.4, we have $\mathbf{y}_{ii}(0) \in [0,1]^{|\mathcal{P}_i|}$, $i \in \mathcal{A}$. Following a stochastic Pipage rounding procedure, each agent $i \in \mathcal{A}$ at each rounding iteration τ randomly selects two fractional elements $[\mathbf{y}_{ii}(\tau)]_p, [\mathbf{y}_{ii}(\tau)]_q$ of $\mathbf{y}_{ii}(\tau)$, i.e., $[\mathbf{y}_{ii}(\tau)]_p, [\mathbf{y}_{ii}(\tau)]_q \in$

Step 1	Randomized selection	0.15	0.25	0.10	0.20	0.10	0.80	0.05	0.35
	Randomized swapping	0	0.25	0.10	0.20	0.25	0.80	0.05	0.35

Step 2	Randomized selection	0	0.25	0.10	0.20	0.25	0.80	0.05	0.35
	Randomized swapping	0	0.25	0.10	0.20	0.25	1	0.05	0.15

Figure 4.1 – The first two steps of the stochastic Pipage rounding (4.48) for an agent i with $\mathbf{x}_{ii}(T) = [0.15, 0.25, 0.1, 0.2, 0.1, 0.8, 0.05, 0.35]^\top$ that should choose two strategies from $\mathcal{P}_i$.

$(0, 1)$, and performs the randomized swapping/update

$$\begin{cases} [\mathbf{y}_{ii}(\tau+1)]_p = [\mathbf{y}_{ii}(\tau)]_p - \delta_p(\tau), \\ [\mathbf{y}_{ii}(\tau+1)]_q = [\mathbf{y}_{ii}(\tau)]_q + \delta_p(\tau), \end{cases} \text{w.p. } \frac{\delta_q(\tau)}{\delta_p(\tau)+\delta_q(\tau)},$$
$$\begin{cases} [\mathbf{y}_{ii}(\tau+1)]_q = [\mathbf{y}_{ii}(\tau)]_q - \delta_q(\tau), \\ [\mathbf{y}_{ii}(\tau+1)]_p = [\mathbf{y}_{ii}(\tau)]_p + \delta_q(\tau), \end{cases} \text{w.p. } \frac{\delta_p(\tau)}{\delta_p(\tau)+\delta_q(\tau)}, \tag{4.48}$$

where $\delta_p(\tau) = \min([\mathbf{y}_{ii}(\tau)]_p, 1-[\mathbf{y}_{ii}(\tau)]_q)$ and $\delta_q(\tau) = \min(1-[\mathbf{y}_{ii}(\tau)]_p, [\mathbf{y}_{ii}(\tau)]_q)$; see Fig. 4.1 for an illustration. Here, 'w.p.' stands for 'with probability of'. The following proposition gives the convergence result of distributed Pipage rounding (4.48).

Proposition 4.2.3. *Starting from $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$, let each agent $i \in \mathcal{A}$ implement the rounding procedure (4.48). Then, $\mathbf{y}_{ii}(|\mathcal{P}_i|) \in \{0,1\}^{|\mathcal{P}_i|}$, and $\mathbf{1}.\mathbf{y}_{ii}(|\mathcal{P}_i|) = \kappa_i$. Moreover, $\bar{\mathbf{y}} = [\mathbf{y}_{11}(|\mathcal{P}_1|), \mathbf{y}_{22}(|\mathcal{P}_2|), \cdots, \mathbf{y}_{NN}(|\mathcal{P}_N|)]$ is a vertex of $\mathcal{M}$.*

Proof. Given the definition of $\delta_p(\tau)$ and $\delta_q(\tau)$, at each iteration of (4.48), either $[\mathbf{y}_{ii}(\tau+1)]_p \in \{0,1\}$ or $[\mathbf{y}_{ii}(\tau+1)]_q \in \{0,1\}$. Moreover, $\mathbf{y}_{ii}(\tau+1) \in [0,1]^{|\mathcal{P}_i|}$. Consequently, $\mathbf{y}_{ii}(|\mathcal{P}_i|) \in \{0,1\}^{|\mathcal{P}_i|}$. Next, note that since $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$, $i \in \mathcal{A}$, by virtue of Lemma 4.2.4, we have $\mathbf{1}.\mathbf{y}_{ii}(0) = \kappa_i$. Therefore, because (4.48) is a zero-sum iteration, we have $\mathbf{1}.\mathbf{y}_{ii}(\tau) = \kappa_i$, $i \in \mathcal{A}$ for any $\tau \in \mathbb{Z}_{\geq 0}$, which confirms $\mathbf{1}.\mathbf{y}_{ii}(|\mathcal{P}_i|) = \kappa_i$ and $\bar{\mathbf{y}} \in \mathcal{M}$. Lastly, because

$[\mathbf{y}_{ii}(|\mathcal{P}_i|)]_r \in \{0,1\},\ r \in \mathcal{P}_i$ for any $i \in \mathcal{A}$, $\bar{\mathbf{y}}$ is a vertex of $\mathcal{M}$. □

It is worth noting that $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i$, guaranteed by Lemma 4.2.4 for our proposed algorithm (4.16), has a significant importance in enabling a rounding procedure without the necessity for coordination among the agents. As we discuss in the numerical example of Section 3.3, $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i$ is not the case for the distributed continuous greedy algorithm of [2], which uses an average consensus algorithm to coordinate the local probability membership choices of the agents.

Our distributed stochastic Pipage rounding procedure concludes by each agent $i \in \mathcal{A}$ choosing its suboptimal strategy set according to

$$\begin{aligned} &\bar{\mathcal{R}}_i = \mathcal{R}_{\bar{\mathbf{y}}_i}, \quad \text{where} \\ &\bar{\mathbf{y}}_i = [\mathbf{0}^\top_{|\mathcal{P}_1|\times 1}, \cdots, \mathbf{y}_{ii}(|\mathcal{P}_i|)^\top, \cdots, \mathbf{0}^\top_{|\mathcal{P}_N|\times 1}]^\top, \end{aligned} \tag{4.49}$$

with $\mathbf{y}_{ii}(|\mathcal{P}_i|)$ obtained from (4.48), initialized at $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$. The following result shows that our proposed distributed rounding procedure (4.48) results in a strategy selection (4.49) that is loss-less in the expected value sense. That is, it results in not only a feasible selected strategy set but also strategies that are selected in a distributed way with no loss in the utility function compared to the fractional solution.

Theorem 4.2.4 (Utility evaluation after distributed Pipage rounding)**.** *Let each agents $i \in \mathcal{A}$ choose its strategy set $\bar{\mathcal{R}}_i \subset \mathcal{P}_i$ according to* (4.49)*. Let $\bar{\mathcal{R}} = \bigcup_{i\in\mathcal{A}} \bar{\mathcal{R}}_i$. Then,*

$$F(\bar{\mathbf{x}}(T)) \leq \mathbb{E}[f(\bar{\mathcal{R}})]. \tag{4.50}$$

Proof. Consider the distributed Pipage rounding (4.48). Let τ_i be any arbitrary iteration stage of (4.48) for agent $i \in \mathcal{A}$. Recall that we partitioned $\mathbf{y}_i$, $i \in \mathcal{A}$ as $\mathbf{y}_i(\tau_i) =$

$[\hat{\mathbf{y}}_{i1}(\tau_i), \cdots, \mathbf{y}_{ii}(\tau_i), \cdots, \hat{\mathbf{y}}_{iN}(\tau_i)]$. Let

$$\mathbf{y}(\tau) = [\mathbf{y}_{11}(\tau_1), \cdots, \mathbf{y}_{ii}(\tau_i + \tau), \cdots, \mathbf{y}_{NN}(\tau_N)]$$

for any $\tau_j \in \mathbb{Z}_{\geq 0}$, $j \in \mathcal{A}$, and arbitrary $i \in \mathcal{A}$.

Distributed Pipage rounding (4.48) results in

$$\mathbf{y}(\tau+1) = \begin{cases} \mathbf{y}(\tau) + \delta_p(\tau_i)\,\mathbf{z}, & \text{w.p. } \frac{\delta_q(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} \in [0,1], \\ \mathbf{y}(\tau) - \delta_q(\tau_i)\,\mathbf{z}, & \text{w.p. } \frac{\delta_p(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} \in [0,1], \end{cases}$$

for a $\mathbf{z} \in \{-1, 0, 1\}^n$ that satisfies $[\mathbf{z}]_p = 1,\ [\mathbf{z}]_q = -1$ and $[\mathbf{z}]_r = 0,\ r \neq p, q$. Next, note that the directional convexity of the multilinear function in Lemma 1.5.6 yields

$$\begin{aligned} F(\mathbf{y}(\tau)) \leq & \frac{\delta_q(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} F(\mathbf{y}(\tau) + \delta_p(\tau_i)\mathbf{z}) + \\ & \frac{\delta_p(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} F(\mathbf{y}(\tau) - \delta_q(\tau_i)\mathbf{z}). \end{aligned}$$

Hence, we can write

$$F(\mathbf{y}(\tau)) \leq \mathbb{E}[F(\mathbf{y}(\tau+1)) \big| \mathbf{y}(\tau)]. \tag{4.51}$$

Next, taking expectation with respect to $\mathbf{y}(\tau)$, we get

$$\mathbb{E}[F(\mathbf{y}(\tau))] \leq \mathbb{E}[F(\mathbf{y}(\tau+1))]. \tag{4.52}$$

Note that because $\mathbf{y}(0)|_{\{\tau_j\}_{j=1}^N = \{\mathbf{0}\}^N} = \bar{\mathbf{x}}(T)$, we have $\mathbb{E}[F(\mathbf{y}(0)|_{\{\tau_j\}_{j=1}^N = \{\mathbf{0}\}^N})] = F(\bar{\mathbf{x}}(T))$. Consequently, since $\mathbf{y}(\tau)$ is defined for any arbitrary $\{\tau_j\}_{j=1}^N$, we can conclude that

$$F(\bar{\mathbf{x}}(T)) \leq \mathbb{E}[F(\bar{\mathbf{y}})], \tag{4.53}$$

where $\bar{\mathbf{y}} = [\mathbf{y}_{11}(|\mathcal{P}_1|), \mathbf{y}_{22}(|\mathcal{P}_2|), \cdots, \mathbf{y}_{NN}(|\mathcal{P}_N|)]$.

Proposition 4.2.3 states that $\bar{y}$ is a vertex of $\mathcal{M}$, therefore $F(\bar{\mathbf{y}}) = f(\mathcal{R}_{\bar{\mathbf{y}}})$. On the other hand, it follows from (4.49) that $\mathcal{R}_{\bar{\mathbf{y}}} = \bigcup_{i\in\mathcal{A}} \bar{\mathcal{R}}_i$. Consequently, (4.50) follows from (4.53). □

4.2.3 A Minimal Information Implementation

Our proposed suboptimal solution to solve problem (4.1) consistes of iterative propagation step (4.16a), and update step (4.16b), which requires local interaction between neighbors. After T steps, once $\mathbf{x}_i(T)$ is obtained, each agent $i \in \mathcal{A}$ computes its suboptimal solution from (4.49) after running Pipage procedure (4.48) locally for at most $|\mathcal{P}_i|$ steps to compute $\mathbf{y}_{ii}(|\mathcal{P}_i|)$. In the propagation step agents should draw K_i samples of $\mathcal{R}_{\mathbf{x}_i} \subset \mathcal{P}$ to compute $\widetilde{\nabla F}(\mathbf{x}_i(t))$. In what follows, by relying on the properties of the updated local copies of the probability membership vector, we outline a minimum information exchange implementation of our distributed solution. The resulted implementation is summarized as the distributed multilinear-extension-based iterative greedy algorithm presented as Algorithm 5.

Definition 4.1. *Given a set $\mathcal{F} \subset \mathcal{P}\times\mathbb{R}$ and a member $(p,\alpha) \in \mathcal{P}\times\mathbb{R}$, we define the addition operator $\oplus$ as $\mathcal{F}' = \mathcal{F}\oplus\{(p,\alpha)\}$ such that*

$$\mathcal{F}' = \begin{cases} \mathcal{F} \cup \{(p,\alpha)\} & \nexists (p,\gamma) \in \mathcal{F}, \\ (\mathcal{F} \setminus \{(p,\gamma)\}) \cup \{(p,\gamma+\alpha)\} & \forall (p,\gamma) \in \mathcal{F}. \end{cases}$$

Given a collection of sets $\mathcal{F}_i \in \mathcal{P}\times\mathbb{R}$, $i \in \mathcal{A}$, we define the max-operation over these collection as $\underset{i\in\mathcal{A}}{\mathrm{MAX}}\ \mathcal{F}_i = \{(u,\gamma) \in \mathcal{P}\times\mathbb{R} | (u,\gamma) \in \bar{\mathcal{F}}$ s.t. $\gamma = \max_{(u,\alpha)\in\bar{\mathcal{F}}} \alpha\}$, where $\bar{\mathcal{F}} = \bigcup_{i\in\mathcal{A}} \mathcal{F}_i$.

We define the local information set of each agent i at time step t as

$$\mathcal{F}_i(t) = \Big\{(p,\alpha) \in \mathcal{P}\times[0,1] \Big| [\mathbf{x}_i(t)]_p \neq 0 \text{ and } \alpha = [\mathbf{x}_i(t)]_p\Big\}. \tag{4.54}$$

Algorithm 5 Discrete distributed implementation of the continuous greedy algorithm.

Init: $\mathcal{F}_1 \leftarrow \emptyset, \cdots, \mathcal{F}_N \leftarrow \emptyset,\ t \leftarrow 1,$
while $t \leq T$ **do**
  for $i \in \mathcal{A}$ **do**
    Draw K_i sample strategy sets, $\{\mathcal{R}_i^k\}_{k=1}^{K_i}$ such that $q \in \mathcal{R}_i^k$ w.p. α for all $(q, \alpha) \in \mathcal{F}_i$.
    for $p \in \mathcal{P}_i$ **do**
      Compute $w_p^i \sim \mathbb{E}[f(\mathcal{R} \cup \{p\}) - f(\mathcal{R} \setminus \{p\})]$ using the strategy sample sets of step 4.
    end for
    $\{p_1^\star, \cdots, p_{\kappa_i}^\star\} = \arg\max\ (\{w_p^i\}_{p \in \mathcal{P}_i}, \kappa_i)$
    $\mathcal{F}_i^- \leftarrow \mathcal{F}_i \oplus \left\{(p_1^\star, \frac{1}{T})\right\} \oplus \cdots \oplus \left\{(p_{\kappa_i}^\star, \frac{1}{T})\right\}$
    Broadcast $\mathcal{F}_i^-$ to the neighbors $\mathcal{N}_i$.
    $\mathcal{F}_i \leftarrow \underset{j \in \mathcal{N}_i \cup \{i\}}{\text{MAX}}\ \mathcal{F}_j^-$
  end for
  $t \leftarrow t + 1.$
end while
for $i \in \mathcal{A}$ **do**
  $\bar{\mathcal{R}}_i = \{\bar{p}_1, \cdots, \bar{p}_{\kappa_i}\} \leftarrow$ `DistStochPipage`$(\mathcal{F}_i)$
end for
Return $\bar{\mathcal{R}}_1, \cdots, \bar{\mathcal{R}}_N$

Since $\mathbf{x}_i(0) = \mathbf{0}$, then $\mathcal{F}_i(0) = \emptyset$. Introduction of the information set $\mathcal{F}_i(t)$ provides a framework through which the agents only store and communicate the necessary information. Furthermore, it enables the agents to carry out their local computations using the available information in $\mathcal{F}_i(t)$.

We start by observing that since in (4.17) we have $\mathbf{w} \in \mathcal{M}_i$, to carry out the propagation step (4.16a), each agent should only compute $[\widetilde{\nabla F}(\mathbf{x}_i(t))]_p$, $p \in \mathcal{P}_i$. At each agent $i \in \mathcal{A}$, define $w_p^i(t) = \left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p$, $p \in \mathcal{P}_i$. Then $w_p^i(t)$, $p \in \mathcal{P}_i$ is computed from

$$w_p^i(t) = \left(\sum_{k=1}^{K_i} f(\mathcal{R}_i^k(t) \cup \{p\}) - f(\mathcal{R}_i^k(t) \setminus \{p\})\right) \Big/ K_i, \tag{4.55}$$

using K_i samples of $\{R_i^k(t)\}_{k=1}^{K_i}$ such that $q \in R_i^k(t)$ with the probability of α for all couples $(q, \alpha) \in \mathcal{F}_i(t)$. Notice that by definition (4.54), $\mathcal{F}_i(t)$ is a set of couples representing which

Algorithm 6 `DistStochPipage( )`

Input: $\mathcal{F}_i$
Init: $\bar{\mathcal{R}}_i = \emptyset$
while $|\bar{\mathcal{R}}_i| < \kappa_i$ **do**
  pick any $(\alpha_p, p), (\alpha_p, p) \in \mathcal{F}_i$ such that $p, q \in \mathcal{P}_i$
  Set: $\begin{cases} \delta_p = \min\{\alpha_p, 1-\alpha_q\} \\ \delta_q = \min\{\alpha_q, 1-\alpha_p\} \end{cases}$
  Update $\begin{cases} \begin{cases} \alpha_p \leftarrow \alpha_p - \delta_p \\ \alpha_q \leftarrow \alpha_q + \delta_p \end{cases} & \text{w.p. } \frac{\delta_q}{\delta_p+\delta_q} \\ \text{or} \\ \begin{cases} \alpha_q \leftarrow \alpha_q - \delta_q \\ \alpha_p \leftarrow \alpha_p + \delta_q \end{cases} & \text{w.p. } \frac{\delta_p}{\delta_p+\delta_q} \end{cases}$
  if $\alpha_p = 1$ then $\bar{\mathcal{R}}_i \leftarrow \bar{\mathcal{R}}_i \cup \{p\}$, $\mathcal{F}_i \leftarrow \mathcal{F}_i \backslash \{(\alpha_p, p)\}$
  if $\alpha_q = 1$ then $\bar{\mathcal{R}}_i \leftarrow \bar{\mathcal{R}}_i \cup \{q\}$, $\mathcal{F}_i \leftarrow \mathcal{F}_i \backslash \{(\alpha_q, q)\}$
end while
Return $\bar{\mathcal{R}}_i$

element $p \in \mathcal{P}$ has an associated non-zero probability membership vector element in $\mathbf{x}_i(t)$.

It follows from submodularity of f that $f(\mathcal{R}_i^k(t) \cup \{p\}) - f(\mathcal{R}_i^k(t)) \backslash \{p\}) \geq 0$. Thus, $w_p^i(t) \geq 0$, $p \in \mathcal{P}_i$. Consequently, one realization of $\widetilde{\mathbf{v}}_i(t)$ of problem (4.17) is $\mathbf{1}_{\{p_1^\star, \cdots, p_{\kappa_i}^\star\}}$, where

$$\{p_1^\star, \cdots, p_{\kappa_i}^\star\} = \arg\max\ (\{w_p^i\}_{p \in \mathcal{P}_i}, \kappa_i). \tag{4.56}$$

Since $\mathbf{1}_{\{p_1^\star, \cdots, p_{\kappa_i}^\star\}}$ is a realization of $\widetilde{\mathbf{v}}_i(t)$, the corresponding realization of propagation rule (4.16a) over the information set $\mathcal{F}_i(t)$ is

$$\mathcal{F}_i^-(t+1) = \mathcal{F}_i(t) \oplus \{(p_1^\star, \frac{1}{T})\} \oplus \cdots \oplus \{p_{\kappa_i}^\star, \frac{1}{T}\}, \tag{4.57}$$

Given Definition 4.1, what $\oplus$ operator does here is to insert $(p_j^\star, \frac{1}{T})$, $j \in \{1, \cdots, \kappa_i\}$ in agent i's information set if there exists no element $(p_j^\star, \alpha)$, $\alpha \neq 0$ in $F_i(t)$; otherwise, operator $\oplus$

pops out $(p_j^\star, \alpha)$ and replaces it with $(p_j^\star, \alpha + \frac{1}{T})$. Therefore, $\mathcal{F}_i^-(t+1)$ is consistent with the realization of $\mathbf{x}_i^-(t+1)$ through the membership probability vector to information set conversion relation (4.54).

Instead of the agents sharing $\mathbf{x}_i^-(t),\ i \in \mathcal{A}$ with their neighbors, they can share their local information set with their neighboring agents and execute a max operation over their local and received information sets as

$$\mathcal{F}_i(t+1) = \underset{j \in \mathcal{N}_i \cup \{i\}}{\mathrm{MAX}} \mathcal{F}_j^-(t+1). \tag{4.58}$$

Consequently, through the membership probability vector to information set conversion relation (4.54), $\mathcal{F}_i(t+1)$ is consistent with a realization of $\mathbf{x}_i(t+1)$.

Finally, given the definition of $\mathcal{F}_i(t)$ in (4.54) and in light of Proposition 4.2.3, the stochastic rounding procedure (4.48) and (4.49) can be implemented according to Algorithm 6.

In light of the discussion above, Algorithm 5 gives our distributed multilinear extension based suboptimal solution for problem (4.1). The following theorem establishes the optimality bound of $f(\bar{\mathcal{R}})$ where $\bar{\mathcal{R}} = \bigcup_{i \in \mathcal{A}} \{\bar{\mathcal{R}}_i\}$ is generated through the decentralized Algorithm 5.

Theorem 4.2.5 (Convergence guarantee and suboptimality gap of Algorithm 5)**.** *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ be normalized, monotone increasing and submodular set function. Let $\mathcal{R}^\star$ to be the optimizer of problem* (4.1)*. Following the distributed Algorithm 5, the admissible strategy set $\bar{\mathcal{R}}$ with probability of at least $1 - 2T\, n\, \mathrm{e}^{-\frac{1}{8T^2}\underline{K}}$, $\underline{K} = \min_{i \in \mathcal{A}} K_i$ satisfies*

$$\mathbb{E}[f(\bar{\mathcal{R}})] \geq \beta\, f(\mathcal{R}^\star),$$

where β is given in (4.23)*.*

Proof. Given that the information set propagation rules (4.56), (4.57), and (4.58) are a

realization of the vector space propagation rules (4.17), (4.16a), and (4.16b), we can conclude that the vector $\mathbf{y} = [\mathbf{y}_1^\top, \cdots, \mathbf{y}_N^\top]^\top$ defined as

$$\begin{cases} [\mathbf{y}]_p = \alpha, & (p, \alpha) \in \mathcal{F}_i(T), \quad p \in \mathcal{P}_i \\ [\mathbf{y}]_p = 0, & \text{Otherwise} \end{cases}$$

is a realization of $\bar{\mathbf{x}}(T)$ and satisfies $F(\bar{\mathbf{x}}(T)) = F(\mathbf{y})$.

Moreover, sampling a single strategy $\bar{p}_i$ according to $\mathbf{y}_i$ out of $\mathcal{P}_i$ is equivalent to sampling rule (4.48). Noting that $\mathbf{y}$ is a realization of $\bar{\mathbf{x}}(T)$, Lemma 6.1.2 and Theorem 4.2.4 leads us to concluding the proof. □

The constant approximation factor β is characterized in terms of the total curvature c of the utility function f. Curvature c represents a measure of the diminishing return of a set function. The curvature of $c = 0$ means that the function is modular, i.e., $f(\{p_1, p_2\}) = f(\{p_1\}) + f(\{p_2\})$, $p_1, p_2 \in \mathcal{P}$. We can see from (4.23) that when $c = 0$, $\beta = 1$, meaning that for modular functions our algorithm can find the optimal solution in finite time. On the other hand, $c = 1$ means that there is at least a member that adds no value to function f in a special circumstance. Whenever the total curvature is not known, it is rational to assume the worst case scenario and set $c = 1$.

Remark 4.2.1 (Extra communication for improved optimality gap)**.** *Replacing the update step* (4.16b) *with* $\mathbf{x}_i(t+1) = \mathbf{y}_i(d(\mathcal{G}))$ *where* $\mathbf{y}_i(0) = \mathbf{x}_i^-(t+1)$ *and*

$$\mathbf{y}_i(m) = \max_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{y}_j(m-1), \quad m \in \{1, \cdots, d(\mathcal{G})\},$$

i.e., starting with $\mathbf{x}_i^-(t+1)$ *and recursively repeating the update step* (4.16b) *using the output of the previous recursion for* $d(\mathcal{G})$ *times, each agent* $i \in \mathcal{A}$ *arrives at* $\mathbf{x}_i(t+1) = \bar{\mathbf{x}}(t+1)$ *(recall Lemma 4.2.3). Hence, for this revised implementation, following the proof of Theorem 6.1.2,*

we observe that (4.25) *is replaced by* $\left|\frac{\partial F}{\partial[\mathbf{x}]_p}(\bar{\mathbf{x}}(t)) - \frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_i(t))\right| = 0$*, which consequently, leads to*

$$\frac{1}{c}(1-\mathrm{e}^{-c})(1-(\frac{c\,\kappa}{2}+1)\frac{\kappa}{T})f(\mathcal{R}^\star) \leq F(\bar{\mathbf{x}}_{ii}(T)), \tag{4.59}$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$*. This improved optimality gap is achieved by* $(d(\mathcal{G})-1)T$ *extra communication per agent. The optimality bound* (4.59) *is the same bound that is achieved by the centralized algorithm of [46]. To implement this revision, Algorithm 5's step 11 (equivalent to* (4.57)*) should be replaced by* $\mathcal{F}_i = \mathcal{H}_i(d(\mathcal{G}))$*, where* $\mathcal{H}_i(0) = \mathcal{F}_i^-$*, and*

$$\mathcal{H}_i(m) = \underset{j\in\mathcal{N}_i\cup\{i\}}{\mathrm{MAX}}\,\mathcal{H}_j^-(m-1), \quad m \in \{1,\cdots,d(\mathcal{G})\}. \tag{4.60}$$

4.3 Numerical Evaluation

We demonstrate our algorithm's performance using a multi-agent information harvesting problem. Consider a countable set of information sources $\mathcal{D} \subset \mathbb{R}^2$ that are spread in a two-dimensional area without any prior information on their spread density function. In the same area, a countable set of prespecified information retrieval points $\mathcal{B} \subset \mathbb{R}^2$ are available for placing information harvester devices. We assume that the information is best transferred from an information point $d \in \mathcal{D}$ to a harvester device dispatched at $b \in \mathcal{B}$ if the distance between b and d is minimized. Hence, for each information point $d \in \mathcal{D}$ the closest information retrieval point $b \in \mathcal{B}$ with a deployed device is assigned to harvest information.

Each agent $i \in \mathcal{A}$ is only able to deploy at most κ_i devices to its admissible deployment locations $\mathcal{B}_i \subset \mathcal{B}$, where $\mathcal{B}_1,\cdots,\mathcal{B}_N$ are not necessarily disjoint sets. To make the strategy set of the agents disjointwe define the deployment strategy of each agent $i \in \mathcal{A}$ as $\mathcal{P}_i =$

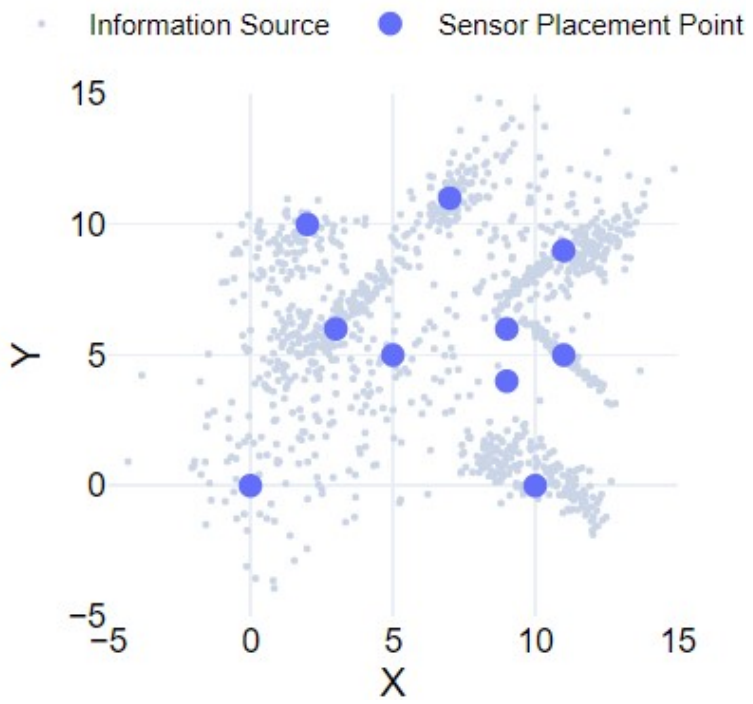


Figure 4.2 – Set of information sources $\mathcal{D}$ and set of sensor placement point $\mathcal{B}$.

$\{(i,b)|b \in \mathcal{B}_i\}$. Note that if $b \in \mathcal{B}_i$ and $b \in \mathcal{B}_j$ then the strategies $(i,b) \in \mathcal{P}_i$ and $(j,b) \in \mathcal{P}_j$ will be placing one sensors from agent $i \in \mathcal{A}$ and one sensor from agent $j \in \mathcal{A}$ at the placement location $b \in \mathcal{B}$. The goal of the agents $\mathcal{A}$ is to each choose a strategy set $\mathcal{R}_i \subset \mathcal{P}_i$, $|\mathcal{R}_i| \leq \kappa_i$ such that cumulative strategy of the team $\mathcal{R} = \bigcup_{i \in \mathcal{A}} \mathcal{R}_i$ results in smallest total distance of information sources to the deployed devices, i.e. minimizing

$$L(\mathcal{R}) = \sum_{d \in \mathcal{D}} \min_{(i,b) \in \mathcal{R}} \|d - b\| . \tag{4.61}$$

Taking a *phantom* placement location b_0 to be a random point in $\mathbb{R}^2$, the problem can be reformulated as problem (4.1) where the utility function to maximize is

$$f(\mathcal{R}) = L(\{b_0\}) - L(\mathcal{R} \cup \{b_0\}). \tag{4.62}$$

This utility function (4.62) measures the decrease in the loss associated with the active set versus the loss associated with just the phantom placement location and maximizing

this function is equivalent to minimizing the loss (4.61). It is known that the utility function (4.62) is submodular and monotone increasing [133].

For our numerical study, we consider 2000 information sources spread in a two-dimensional field where there are 10 deployment locations $\mathcal{B} = \{b_1, \cdots, b_{10}\}$, see Fig. 4.2. We consider a set of five agents $\mathcal{A} = \{1, 2, 3, 4, 5\}$ whose goal is to deploy $\kappa_1 = 5,\ \kappa_2 = 2,\ \kappa_3 = 1,\ \kappa_4 = 1,\ \kappa_5 = 1$ devices at $\mathcal{B}_1 = \{b_1 \cdots, b_{10}\}$, $\mathcal{B}_2 = \{b_1, \cdots, b_5\}$, $\mathcal{B}_3 = \{b_1, b_2, b_3\}$, $\mathcal{B}_4 = \{b_1, b_2\}$, and $\mathcal{B}_5 = \{b_1, b_2\}$. Hence the disjoint strategy sets are defined as $\mathcal{P}_1 = \{(1, b_1), \cdots, (1, b_{10})\}$, $\mathcal{P}_2 = \{(2, b_1), \cdots, (2, b_{10})\}$, $\mathcal{P}_3 = \{(3, b_1), (3, b_2), (3, b_3)\}$, $\mathcal{P}_4 = \{(4, b_1), (4, b_2)\}$, and $\mathcal{P}_5 = \{(5, b_1), (5, b_2)\}$. Although, the general form of the problem is NP-hard, we have designed our numerical example such the optimal solution is trivial. Recall that to maximize the utility (4.62) of the group the deployed sensors must be placed such that the distance between the information sources and deployed sensors is minimized. Since there are $|\mathcal{B}| = 10$ deployment locations and $\sum_{j=1}^{5} \kappa_j = 10$ sensors to deploy, the optimal solution is to place the deployed sensors to occupy all the sensor-placement locations. This deployment scenario is only feasible if agent 1 deploys its $\kappa_1 = 5$ devices at locations $\{b_6, \cdots, b_{10}\} \subset \mathcal{B}_1$, i.e. $\mathcal{R}_1 = \{(1, b_6), (1, b_7), (1, b_8), (1, b_9), (1, b_10)\}$, agent 2 placing its $\kappa_2 = 2$ at $\{b_4, b_5\} \subset \mathcal{B}_2$, i.e. $\mathcal{R}_2 = \{(2, b_4), (2, b_5)\}$, agent 3 placing its $\kappa_3 = 1$ at $b_3 \in \mathcal{B}_3$, i.e. $\mathcal{R}_3 = \{(3, b_3)\}$, agent 4 placing its $\kappa_4 = 1$ at $b_2 \in \mathcal{B}_4$, i.e. $\mathcal{R}_4 = \{(4, b_2)\}$, and agent 5 placing its $\kappa_5 = 1$ at $b_1 \in \mathcal{B}_5$, i.e. $\mathcal{R}_5 = \{(5, b_1)\}$. This setting allows us to compare the outcome of the suboptimal solutions against the optimal one. It is interesting to notice that the total curvature of utility function (4.62) is $c = 1$. This is because if we take the strategy set $\mathcal{S} = \{(1, b_1), (2, b_2), (3, b_3)\}$ and the strategy $p = (4, b_1)$, since, the strategies $(1, b_1)$ and $(4, b_1)$ place a sensor at the same location then the utility equation (4.62) results in $\Delta_f(p|\mathcal{S}) = 0$. Thus, given the definition of the total curvature (4.21), we obtain $c = 1$.

Let the communication topology of the agents be an undirected ring graph, see Fig. 4.3. First, we solve the problem using Algorithm 5. To evaluate the performance of our algorithm we

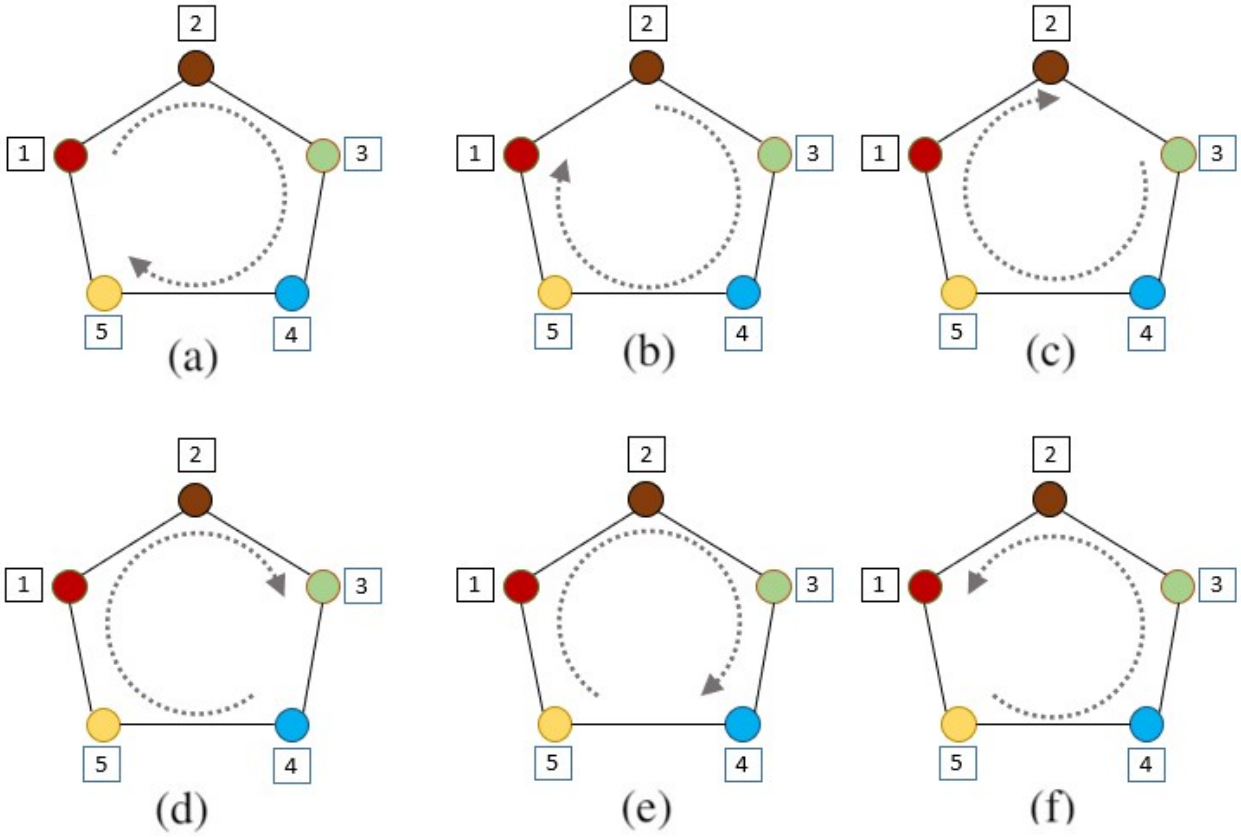


Figure 4.3 – The communication graph of the agents is a ring graph. Six possible communication sequences to implement a sequential greedy algorithm are shown.

generate 50 deployment scenarios, each corresponding to a set of 2000 randomly generated information sources and 10 sensor locations. The results of implementing Algorithm 5 for the different number of samples K_i (all agents use the same number of samples) and iteration number $T = 50$ is shown in Fig. 4.4. Observe that using a modest number of iterations $T = 50$ and a modest number of samples $K_i = 1000$ Algorithm 5 finds almost the optimal solution in terms of occupying the placement locations; the average number of locations occupied is 9.8. For this setting,the expected outcome of Algorithm 5 over the 50 placement scenarios we consider, measured by utility function (4.62), is at 0.95 of the optimal solution. The run-time of the algorithm for each agent is approximately 25 seconds on a computing device with Intel(R) Xeon(R) CPU @ 2.30GHz and 13GB RAM.

Comparison with sequential greedy algorithm: Next, we solve the problem using a decentral-

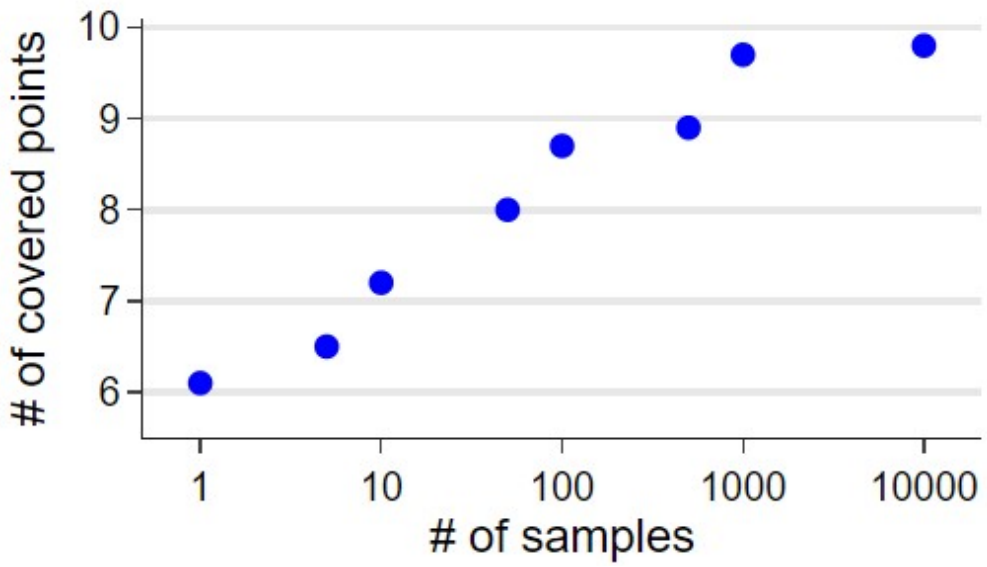


Figure 4.4 – The average number of sensor placement points covered by deployed sensors when Algorithm 5 is implemented by different sample numbers and $T = 50$.

ized *massage-passing* sequential greedy algorithm following [38, 64]. That is, we choose a route `SEQ` that visits all the agents on the communication graph. We then make the agents perform the sequential greedy algorithm by sequential message-passing according to `SEQ`. Fig. 4.3(a)-(f) gives 6 of the possible `SEQ` depicted by the semi-circular arrow inside the networks. As Fig. 4.5 shows the performance (measured by the number of occupied placement locations) of the sequential greedy algorithm depends on what `SEQ` agents follow, with `SEQ` of Fig. 4.3(a) delivering the worst performance. Moreover, the performance measured by utility function (4.62) for `SEQ` (a),(b),(c),(d),(e), and (f) are respectively 0.75, 0.81, 0.87, 0.91, 0.85, 0.99 of the optimal utility value. We can attribute this inconsistency to the heterogeneity of the agents' sensor numbers. When agents with a larger number of choices pick first, this limits the options of the agents with a lower number of sensors available. However, the performance of Algorithm 5 is regardless of any particular path on the graph since, through its iterative process, the agents get the chance to readjust their choices, see Fig. 4.6 for a deployment outcome via Algorithm 5 and the sequential greedy algorithm. Intuitively, this explains the better optimality gap of the continuous greedy algorithm over the sequential greedy algorithm. The sequential greedy algorithm has a run-time of less than 1 second

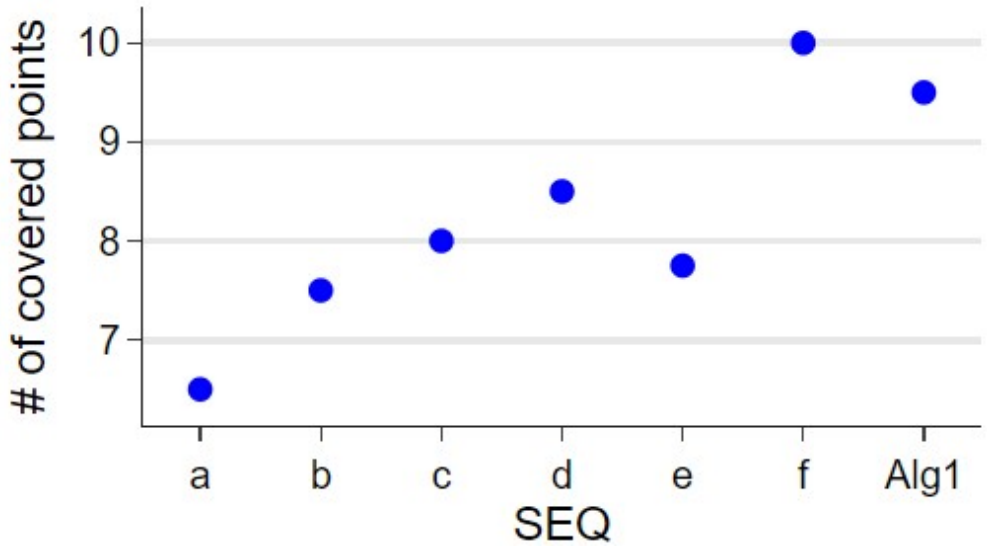


Figure 4.5 – The average number of covered placement points over 50 different randomly generated information sources and sensor placement locations. The x-axis corresponds to the six `SEQ` in Fig. 4.3(a)-(f) and Algorithm 5 denoted by Alg1. The y-axis corresponds to the average number of sensor placement points covered by the deployed sensors.

for each agent on a device with Intel(R) Xeon(R) CPU @ 2.30GHz and 13GB RAM, which is significantly less than 25 seconds that we reported for our proposed Algorithm 5 using $K_i = 1000$ and $T = 50$. Even though computationally efficient, as we discussed in the introduction, the downsides of the sequential greedy algorithm are in its worse optimality gap, the overhead associated with identifying the message-passing sequence, and the dependence of the results on the message-passing sequence [51]. As we discussed earlier, finding the communication sequence resulting in maximum utility value is an NP-Hard problem. To find the best sequence `SEQ` over a communication network to get the highest utility, all possible communication routes between the agents should be examined. When there are no assumptions on the topology of the communication graph, $O(n!)$ is the order of the worst case number of sequences to be examined. In our particular example, the simple structure of the communication graph (a ring) allows us to find the sequence with the highest reward in $O(n)$ search time. Moreover, since our efforts are toward decentralization, a prior decentralized sequence selection algorithm should be designed to synchronize the agents on the sequence the want to examined. This has been further studied in [51].

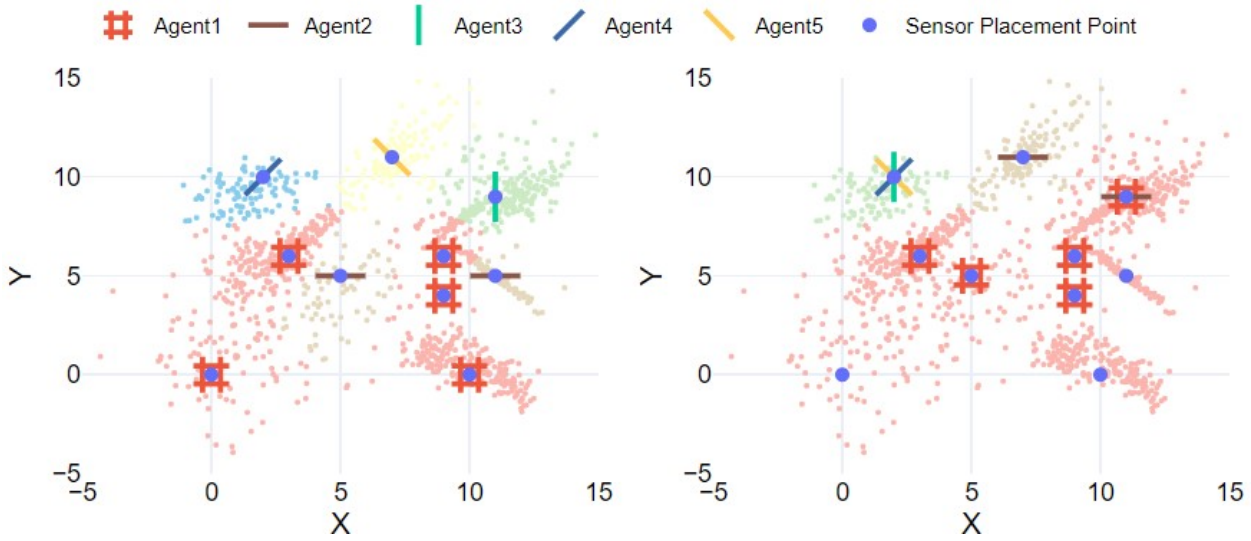


Figure 4.6 – The left figure shows an instance of the placement result for Algorithm 5 and the right figure shows the placement results for the Sequential Greedy of SEQ (a). Algorithm 5 is able to place the sensors such that all of the placement locations are occupied while the Sequential Greedy of SEQ (a) leaves out three unoccupied placement locations.

Comparison with the average consensus based algorithm of [2] : we compare our proposed Algorithm 5, which is based on a maximum consensus communication to the algorithm proposed in [2], which is based on an average consensus communication. Since the algorithm of [2] is designed for when agents choose only one strategy each, we carry out this study for $\kappa_i = 1$ for $i \in \{1, 2, \cdots, 5\}$. The average consensus-based algorithm of [2] is only a distributed implementation of the continuous greedy algorithm. Our analysis shows that for a given number of iteration $T = 50$ and a given number of samples $K_i = 1000$ (same for all agents) the algorithm in [2] yields the local probability vectors $\mathbf{x}_i(T)$ for each agent $i \in \mathcal{A}$ such that $F(\mathbf{x}_i(T)) \geq 0.91 f(\mathcal{R}^\star)$ and also Algorithm 5 yields a probability membership $\bar{\mathbf{x}}(T)$ such that $F(\bar{\mathbf{x}}(T)) \geq 0.92 f(\mathcal{R}^\star)$ (The optimal solution was computed by the brute force search). The main difference between the two algorithms however is in how $\mathbf{x}_i(T)$ of each agent $i \in \{1, 2, \cdots, 5\}$ is placed with respect to the edges of the matroid polytope $\mathcal{M}$. Recall that $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i = 1$ is of great importance for the rounding procedure. Let

$$D(t) = \sum_{i \in \mathcal{A}} (\mathbf{1}.\mathbf{x}_{ii}(t) - 1) \mathbb{1}((\mathbf{1}.\mathbf{x}_{ii}(t) - 1) \geq 0), \tag{4.63}$$

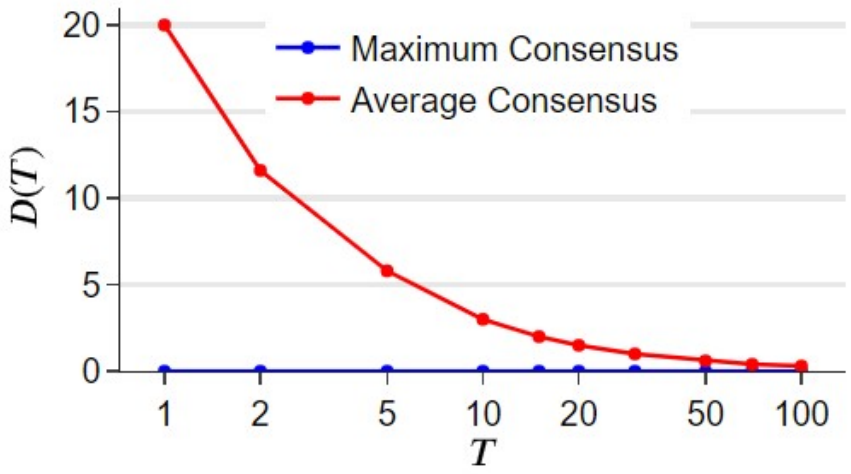


Figure 4.7 – The deviation of probability vectors $\mathbf{x}_i(T)$, $i \in \mathcal{A}$ from the convex hull $\mathcal{M}$ for average consensus and maximum consensus communication protocols.

where $\mathbb{1} : \mathbb{R} \to \{0, 1\}$ is the indicator function. The value of $D(T)$ shows the deviation of the local component of the membership probability vector $\mathbf{x}_{ii}(T)$ from the edges of matroid polytope $\mathcal{M}$. Figure 4.7 shows this value for the different numbers of iterations for the two algorithms; Algorithm 5, as ensured by Lemma 4.2.4, results in $D(T) = 0$ for any choice of iteration number T. However, this is not the case for Algorithm of [2]. As the results in Figure 4.7 shows this algorithm seems to satisfy $D(T) = 0$ as the number of the iteration increases. Figure 4.8 compares the value of $D(t)$ of Algorithm 5 and the algorithm of [2] for $t \in [0, T]$ over 10 different instances of the utility maximization problem (4.62). The former algorithm is based on maximum consensus and $D(t)$ for this algorithm converges to 0, as predicted by Lemma 4.2.4. The latter algorithm is based on average consensus and $D(t)$ converges to a number greater than 0, meaning that $\mathbf{x}_i(T)$ of some of the agents are outside $\mathcal{M}$.

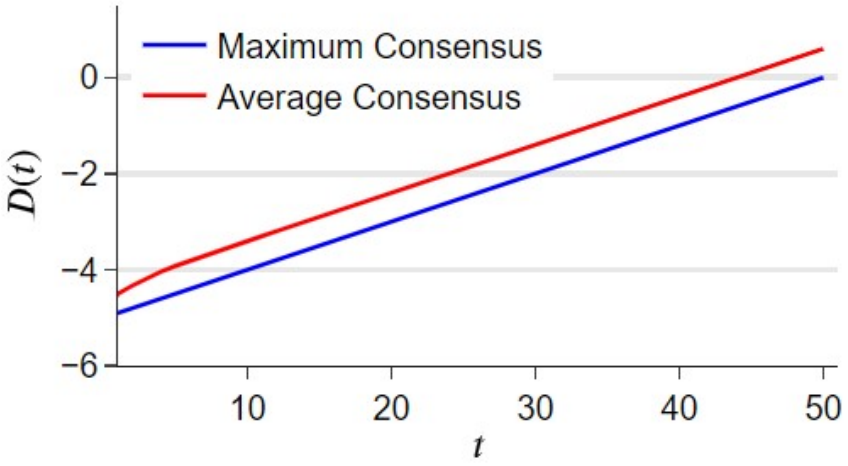


Figure 4.8 – The value of $D(t)$ calculated using $\mathbf{x}_i(t)$, $i \in \mathcal{A}$ for average consensus and maximum consensus communication protocols. The value of $D(t)$ was calculated by running Algorithm 5 and the algorithm in [2] with $\kappa_i = 1,\ i \in \mathcal{A}$ over 10 different instances of the utility maximization problem (4.62).

4.4 Conclusion

We proposed a distributed suboptimal algorithm to solve the problem of maximizing a monotone increasing submodular set function subject to a partition matroid constraint when agents communicate over a connected graph. Our problem of interest was motivated by optimal multi-agent sensor placement problems in discrete space. Our algorithm was a practical decentralization of a multilinear extension-based algorithm that achieves $\frac{1}{c}(1-\mathrm{e}^{-c}-O(1/T))$ optimally gap, which is an improvement over $\frac{1}{1+c}$ optimality gap that the well-known sequential greedy algorithm achieves. Our algorithm included a distributed continuous greedy algorithm followed by a local rounding procedure that required no inter-agent communication. Through a numerical study, we compared the outcome obtained by our proposed algorithm with a decentralized sequential greedy algorithm that is constructed from assigning a priority sequence to the agents. We showed that the outcome of the sequential greedy algorithm is inconsistent and depends on the sequence. However, our algorithm's outcome, due to its iterative nature intrinsically tended to be consistent, which also explains its better optimally gap over the sequential greedy algorithm. We also compared our algorithm to an

existing distributed average consensus-based continuous greedy algorithm. We showed that the main advantage of our proposed algorithm is its strong guarantee of reaching the edges of the constraint set's matroid polytope by all agents in finite time, which is of significance in the Pi-page type rounding procedures. Our future work is to study the robustness of our proposed algorithm to message dropout.

Chapter 5

Private Strategy Selection

This chapter considers a multi-agent submodular set function maximization problem subject to partition matroid in which the utility is shared, but the agents' policy choices are constrained locally. The chapter's main contribution is to design a distributed algorithm that enables each agent to find a suboptimal policy locally with a guaranteed level of privacy. The submodular set function maximization problems are NP-hard. For agents communicate over a connected graph, therefore, this chapter proposes a polynomial-time distributed algorithm to obtain a suboptimal solution with guarantees on the optimality bound. The proposed algorithm is based on a distributed randomized gradient ascent scheme built on the multilinear extension of the submodular set function in the continuous domain. Our next contribution is the design of a distributed rounding algorithm without further communication between the agents. We base our algorithm's privacy preservation characteristic on our proposed stochastic rounding method and tie the level of privacy to the variable $\gamma \in [0, 1]$. That is, the policy choice of an agent can be determined with the probability of at most γ. We show that our distributed algorithm results in a strategy set that when the team's objective function is evaluated at worst case, the objective function value is in $1 - (1/\mathrm{e})^{h(\gamma)} - O(T)$ of the optimal solution, highlighting the interplay between level of optimality gap and guaranteed

level of privacy.

5.1 Problem Statement

We consider a group of $\mathcal{A}$, $|\mathcal{A}| = N$ agents with communication and computation capabilities, interacting over a connected undirected graph $\mathcal{G}(\mathcal{A}, \mathcal{E})$. Each agent $a \in \mathcal{A}$ has a distinct discrete policy set $\mathcal{P}_a$ and wants to choose $\kappa_a \in \mathbb{Z}_{\geq 1}$ policies from its policy set such that a monotone increasing and submodular utility function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, $\mathcal{P} = \bigcup_{a \in \mathcal{A}} \mathcal{P}_a$, evaluated at all the agents' policy selection is maximized[1]. In other words, the agents aim to solve in a distributed manner the optimization problem

$$\max_{\mathcal{R} \subset \mathcal{P},\, \mathcal{R} \in \mathcal{I}} f(\mathcal{R}) \quad \text{where} \tag{5.1a}$$

$$\mathcal{I} = \big\{ \mathcal{S} \subset \mathcal{P} \,\big|\, |\mathcal{S} \cap \mathcal{P}_a| \leq \kappa_a,\ a \in \mathcal{A} \big\}. \tag{5.1b}$$

Agents' access to the utility function is through a black box that returns $f(\mathcal{R})$ for any given set $\mathcal{R} \in \mathcal{P}$ (value oracle model). Constraint set (5.1b) is the *partition matroid*, which restricts the number of policy choices of each agent $a \in \mathcal{A}$ to a prespecified number κ_a. While seeking a distributed solution for (5.1), each agent wants to have a formal guarantee that its final policy choice stays private. Even though a distributed solution eliminates the necessity of information aggregation in a central location, inter-agent communication can still expose distributed network operations to adversarial eavesdroppers. These adversaries can be other agents in the network or outside eavesdroppers that intercept communication messages. Because in problem (5.1) the agents have joint utility function, privacy preservation is particularly a challenging problem.

Definition 2. *Given a distributed algorithm used by agent $a \in \mathcal{A}$ to solve the policy selection*

[1]For clarity, we provide a brief description of the notation and the definitions in Section **??**.

problem 4.1 to achieve the policy set $\bar{\mathcal{R}}_a \in \mathcal{P}_a$ is γ-Private where $\gamma \in [0,1]$, if an intelligent entity other than agent a is only able to estimate $p \in \bar{\mathcal{R}}_a$ with probability of at most $1-\gamma$, for all $p \in \mathcal{P}_a$

A distributed solution to problem (5.1) should enable each agent $a \in \mathcal{A}$ to choose κ_a policies from its local policy set $\mathcal{P}_a$. To propose our distributed algorithm, we assert a problem reformulation by introducing virtual agents to the network. We split each agent $a \in \mathcal{A}$ into κ_a fully connected local sub-agents embedded in agent a, see Fig 5.1, and design the distributed algorithm such that each sub-agent is responsible to choose a single policy. Without loss of generality, we index the sub-agent set as $\mathcal{A}^{\text{Ex}} = \{1, \cdots, N^{\text{Ex}}\}$ where $N^{\text{Ex}} = \sum_{a\in\mathcal{A}} \kappa_a$ and $\mathcal{A}_a = \{a_1, \cdots, a_{\kappa_a}\} \subset \mathcal{A}^{\text{Ex}}$. Hence $\mathcal{A}^{\text{Ex}} = \bigcup_{a\in\mathcal{A}} \mathcal{A}_a$. Moreover, we define the ground set in the extended space as $\mathcal{P}^{\text{Ex}} = \{1, \cdots, n^{\text{Ex}}\}$, where $n^{\text{Ex}} = \sum_{a\in\mathcal{A}} \kappa_a |\mathcal{P}_a|$ and the policy set of each sub-agent $i \in \mathcal{A}_a \subset \mathcal{A}^{\text{Ex}}$ is defined as $\mathcal{P}_i^{\text{Ex}} = \{p_i^1, \cdots, p_i^{|\mathcal{P}_a|}\} \subset \mathcal{P}^{\text{Ex}}$ and $\mathcal{P}^{\text{Ex}} = \bigcup_{i\in\mathcal{A}^{\text{Ex}}} \mathcal{P}_i^{\text{Ex}}$. The policy set of sub-agent $i \in \mathcal{A}_a$ is defined as a copy of the policy set of the agent a. Hence, given that $p_i^l \in \mathcal{P}_i^{\text{Ex}}$ and $p_j^l \in \mathcal{P}_j^{\text{Ex}}$ are the copies of the l-th policy of agent $a \in \mathcal{A}$ where sub-agents $i, j \in \mathcal{A}_a$ then for any $\mathcal{R} \in \mathcal{P}^{\text{Ex}}$ the following holds

$$\Delta_f(p_i^l|\mathcal{R}) = \Delta_f(p_j^l|\mathcal{R}) \tag{5.2a}$$

$$\Delta_f(p_i^l|\mathcal{R} \cup \{p_j^l\}) = \Delta_f(p_j^l|\mathcal{R} \cup \{p_i^l\}) = 0 \tag{5.2b}$$

Moreover, we define the policy mapping function $\mathsf{PolicMap}(p) = q,\ p \in \mathcal{P}_i^{\text{Ex}}$ to return $q \in \mathcal{P}_a$ where p is a copy of policy q. In this extended space, problem (5.1) can equivalently be represented as

$$\max_{\mathcal{R}\subset\mathcal{P}^{\text{Es}},\, \mathcal{R}\in\mathcal{I}^{\text{Ex}}} f(\mathcal{R}) \quad \text{s.t.} \tag{5.3a}$$

$$\mathcal{I}^{\text{Ex}} = \big\{\mathcal{R} \subset \mathcal{P}^{\text{Ex}} \,\big|\, |\mathcal{R} \cap \mathcal{P}_i^{\text{Ex}}| \leq 1,\ i \in \mathcal{A}^{\text{Ex}}\big\}. \tag{5.3b}$$

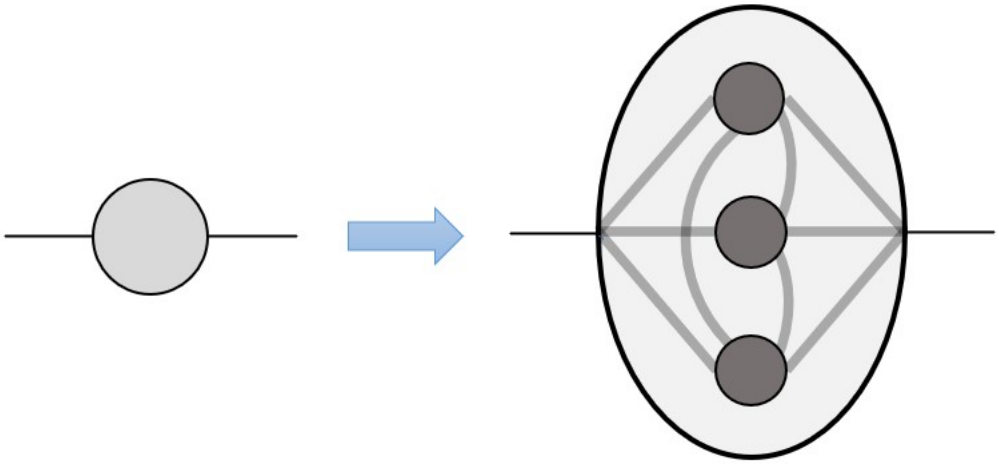


Figure 5.1 – The extension of an agent with $\kappa_i = 3$ to the sub-agents.

Without loss of generality we assume that $\mathcal{P}^{\text{Ex}}$ is ordered according to $\mathcal{A}^{\text{Ex}}$ in a sense that $1 \in \mathcal{P}_1^{\text{Ex}}$ and $n^{\text{Ex}} \in \mathcal{P}_{N^{\text{Ex}}}^{\text{Ex}}$. $\mathcal{A}^{\text{Ex}}$ is ordered in a way that sub-agents of an agent are ordered sequentially and the sub-agent sets are ordered in accordance with their corresponding agent order in $\mathcal{A}$. With the new formulation of the problem in the context of sub-agents, the optimal solution to problem (5.3) is equivalent to the optimal solution of the main problem (5.1). The equivalent problem formulation suggests that instead of each agent $a \in \mathcal{A}$ selecting κ_a policies, they create κ_a sub-agents and each selecting only one policy out of $\mathcal{P}_a$.

To solve (5.3), we use a continuous relaxation method. Notice that the utility set function f assigns values to all the subsets of $\mathcal{P}^{\text{Ex}} = \bigcup_{i \in \mathcal{A}^{\text{Ex}}} \mathcal{P}_i^{\text{Ex}} = \{1, \cdots, n^{\text{Ex}}\}$. Thus, equivalently, we can regard the set value utility function as a function on the Boolean hypercube $\{0,1\}^{n^{\text{Ex}}}$, i.e., $f : \{0,1\}^{n^{\text{Ex}}} \to \mathbb{R}$. For a submodular function $f : 2^{\mathcal{P}^{\text{Ex}}} \to \mathbb{R}_{\geq 0}$, its multilinear-extension $F : [0,1]^n \to \mathbb{R}_{\geq 0}$ in the continuous space is [4]

$$F(\mathbf{x}) = \sum_{\mathcal{R} \subset \mathcal{P}^{\text{Ex}}} f(\mathcal{R}) \prod_{p \in \mathcal{R}} [\mathbf{x}]_p \prod_{p \notin \mathcal{R}} (1 - [\mathbf{x}]_p), \quad \mathbf{x} \in [0,1]^{n^{\text{Ex}}}, \tag{5.4}$$

which expands the function evaluation of the utility function over the space between the vertices of the Boolean hypercube $\{0,1\}^{n^{\text{Ex}}}$. Given $\mathbf{x} \in [0,1]^{n^{\text{Ex}}}$ we can define $\mathcal{R}_{\mathbf{x}}$ to be the random subset of $\mathcal{P}$ in which each element $p \in \mathcal{P}$ is included independently with probability

$[\mathbf{x}]_p$ and not included with probability $1-[\mathbf{x}]_p$. Then the multilinear-extension F in (5.4) is interpreted

$$F(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_{\mathbf{x}})], \tag{5.5}$$

where $\mathbb{E}[.]$ indicates the expected value. Then, we obtain [4]

$$\frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_{\mathbf{x}} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}} \setminus \{p\})]. \tag{5.6}$$

The partition matroid constraint is also extended to continuous space using the *matroid polytope*

$$\mathcal{M} = \big\{\mathbf{x} \in [0,1]^{n^{\mathrm{Ex}}} \;\big|\; \sum\nolimits_{p \in \mathcal{P}_i^{\mathrm{Ex}}} [\mathbf{x}]_p \leq 1, \forall i \in \mathcal{A}^{\mathrm{Ex}}\big\}, \tag{5.7}$$

which is the convex hull of the vertices of the hypercube $\{0,1\}^{n^{\mathrm{Ex}}}$ that satisfies the partition matroid constraint (5.3b). Additionally, note that according to (5.4), $F(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{M}$ is a weighted average of values of F at the vertices of the matriod polytope $\mathcal{M}$. Then, equivalently, $F(\mathbf{x})$ at any $\mathbf{x} \in \mathcal{M}$ is a normalized-weighted average of f on the strategies satisfying constraint (5.1b). As such,

$$f(\mathcal{R}^\star) \geq F(\mathbf{x}), \;\; \mathbf{x} \in \mathcal{M}, \;\; \text{and} \;\; f(\mathcal{R}^\star) = F(\mathbf{1}_{\mathcal{R}^\star}),$$

which is equivalent to $f(\mathcal{R}^\star) = \max\limits_{\mathbf{x} \in \mathcal{M}} F(\mathbf{x})$, where $\mathcal{R}^\star$ is the optimizer of problem (5.3) [4]. Therefore, to find $\mathcal{R}^\star$, we can solve the continuous domain optimization problem [4]

$$\max_{\mathbf{x} \in \mathcal{M}} F(\mathbf{x}). \tag{5.8}$$

5.2 Distributed Private Policy Selection

Continuous relaxation (5.8) of the set value optimization problem (5.3) presents us with the plethora of continuous domain optimization solvers such a gradient-based algorithms. But since problem (5.8) is not a concave problem, a gradient accent approach does not necessarily lead to the optimal value. Never the less in this section, we propose Algorithm 7 as a γ-private distributed gradient ascent algorithm base on (5.8) to find a sub-optimal solution to the problem (5.1).

Let every sub-sub-agent $i \in \mathcal{A}^{\mathrm{Ex}}$ maintain and evolve a belief state vector as $\mathbf{x}_i(t) \in \mathbb{R}^{n^{\mathrm{Ex}}}$. Each entry $[\mathbf{x}_i(t)]_p$, $p \in \mathcal{P}_j^{\mathrm{Ex}}$, $j \in \mathcal{A}^{\mathrm{Ex}}$ of the belief state corresponds to the estimate of sub-agent i on the confidence of sub-agent j about choosing p as the final selected policy. Since $\mathcal{P}^{\mathrm{Ex}} = \{1, \cdots, n^{\mathrm{Ex}}\}$ is sorted sub-agent-wise, we denote $\mathbf{x}_i(t) = [\hat{\mathbf{x}}_{i1}^\top(t), \cdots, \mathbf{x}_{ii}^\top(t), \cdots, \hat{\mathbf{x}}_{iN^{\mathrm{Ex}}}^\top(t)]^\top \in \mathbb{R}^{n^{\mathrm{Ex}}}$ where $\mathbf{x}_{ii}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_i^{\mathrm{Ex}}|}$ is the belief vector of sub-agent i's own policy with entries of $[\mathbf{x}_i(t)]_p$, $p \in \mathcal{P}_i^{\mathrm{Ex}}$ at iteration $t \in \{0, 1, \cdots, T\}$, $T \in \mathbb{Z}_{>0}$, while $\hat{\mathbf{x}}_{ij}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_j^{\mathrm{Ex}}|}$ is the local estimate of the belief vector of sub-agent j by sub-agent i with entries of $[\mathbf{x}_i(t)]_p$, $p \in \mathcal{P}_j^{\mathrm{Ex}}$, $j \in \mathcal{A}^{\mathrm{Ex}} \setminus \{i\}$. Let

$$\lambda = \frac{T}{1 - \sqrt[\kappa_{\max}]{1-\gamma}}, \quad \kappa_{\max} = \max_{a \in \mathcal{A}} \kappa_a. \tag{5.9}$$

Every sub-agent $i \in \mathcal{A}^{\mathrm{Ex}}$ initializes at $\mathbf{x}_i(0) = \mathbf{0}$ and implements the *propagation* and *update* steps

$$\mathbf{x}_i^-(t+1) = \mathbf{x}_i(t) + \frac{1}{\lambda}\widetilde{\mathbf{v}}_i(t), \tag{5.10a}$$

$$\mathbf{x}_i(t+1) = \max_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{x}_j^-(t+1), \tag{5.10b}$$

where

$$\widetilde{\mathbf{v}}_i(t) = \underset{\mathbf{w}\in\mathcal{M}_i}{\operatorname{argmax}}\, \mathbf{w}.\widetilde{\nabla F}(\mathbf{x}_i(t)) \tag{5.11}$$

$$\mathcal{M}_i = \Big\{ \mathbf{w}\in[0,1]^{n^{\mathrm{Ex}}} \Big| \, \mathbf{1}^\top.\mathbf{w} \leq 1 \ , [\mathbf{w}]_p = 0, \ \forall p \in \mathcal{P}^{\mathrm{Ex}} \backslash \mathcal{P}_i^{\mathrm{Ex}} \Big\}. \tag{5.12}$$

The value of $\widetilde{\nabla F}(\mathbf{x}_i(t))$ is the empirical estimate of $\nabla F(\mathbf{x}_i(t))$ calculated locally by each agent by using using K_i samples. The distributed sampling and empirical calculation of $\widetilde{\nabla F}(\mathbf{x}_i(t))$ are the same as the method introduced in previous chapter and are omitted here for brevity. In the propagation step (5.10a) sub-agent i takes a step along a feasible gradient ascent direction in its own local polytope (5.12). But because the propagation is only based on the local information, in the update step (5.10b), the propagated $\mathbf{x}_i^-(t+1)$ of each sub-agent $i \in \mathcal{A}$ is updated by element-wise maximum seeking among its neighbors. Because f is monotone increasing, we have $\frac{\partial F}{\partial [\mathbf{x}]_p} \geq 0$, which leads to $\mathbf{1}.\tilde{\mathbf{v}}_i(t) = 1$ where $\tilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$. Therefore, given λ by equation (5.9), at the end of the propagation and update process at time T, we have

$$\mathbf{1}^\top.\mathbf{x}_{ii}(T) = 1 - \sqrt[\kappa_{\max}]{1-\gamma}. \tag{5.13}$$

For the rounding procedure it is essential that each sub-agent has a local belief state vector on its own policy set that sums up to 1. Therefore, given (5.13), each sub-agent $i \in \mathcal{A}^{\mathrm{Ex}}$ applies a γ-private operation on its local belief state vector and generates the local belief vector $\mathbf{z}_i$ such that

$$\mathbf{z}_i = [\mathbf{0}^\top, \cdots, \mathbf{x}_{ii}^\top(T) + \mathbf{y}_{ii}, \cdots, \mathbf{0}^\top]^\top \in \mathbb{R}^{n^{\mathrm{Ex}}} \tag{5.14}$$

where $\mathbf{y}_{ii}$ is generated privately by sub-agent i such that

$$\mathbf{1}^{\top}.\mathbf{y}_{ii} = \sqrt[\kappa_{\max}]{1-\gamma}. \tag{5.15}$$

We are now ready to propose our distributed rounding scheme that enables each agent $a \in \mathcal{A}$ to choose its local policy set. Our proposed rounding scheme does not require any inter-agent communication because our design process leads to $\mathbf{1}^{\top}.(\mathbf{x}_{ii}(T) + \mathbf{y}_{ii}) = 1$ and consequently $\mathbf{1}^{\top}.\mathbf{z}_i = 1$. Our proposed rounding method is cooperative in the level of sub-agents of an agent $a \in \mathcal{A}$. This is an acceptable assumption since the sub-agents of each agent $a \in \mathcal{A}$ are virtual agents created by agent a. Given that $\mathcal{A}_a = \{a_1, \cdots, a_{\kappa_a}\} \subset \mathcal{A}^{\text{Ex}}$, and defining from $\mathcal{T}(0) = \emptyset$, each sub-agent $i = a_k \in \mathcal{A}_a$ of agent $a \in \mathcal{A}$, uses the vector $\mathbf{z}_i$ and a randomly generated variable $\zeta \in [0, 1]$ to choose a single policy $p \in \mathcal{P}_i^{\text{Ex}} \subset \mathcal{P}^{\text{Ex}}$ satisfying

$$\sum\nolimits_{l=1}^{p} [\mathbf{z}_i]_l \leq \zeta \leq \sum\nolimits_{l=1}^{p+1} [\mathbf{z}_i]_l \tag{5.16}$$

to get

$$\mathcal{T}(k) = \mathcal{T}(k-1) \cup \{p\}. \tag{5.17}$$

Setting $\bar{\mathcal{T}}_a = \mathcal{T}(\kappa_a)$, each agent $a \in \mathcal{A}$ chooses its policy set from the policy choices of its sub-agents as

$$\bar{\mathcal{R}}_a = \{\mathsf{PolicyMap}(p) | p \in \mathcal{T}_a\} \tag{5.18}$$

Defining $\bar{\mathcal{R}} = \bigcup_{a \in \mathcal{A}} \bar{\mathcal{R}}_a$ to be the collective selected policies of the agents $\mathcal{A}$, the following theorems assert the guaranteed optimality bound and privacy preservation guarantee of Algorithm 7.

Theorem 5.2.1 (Sub-optimality gap of Algorithm 7)**.** *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ be normalized,*

monotone increasing and submodular set function. Let $\mathcal{R}^\star$ to be the optimizer of problem (5.1)*. Then, the policy set $\bar{\mathcal{R}}$, the output of distributed Algorithm 7, satisfies*

$$\alpha\big(1-\frac{1}{\mathrm{e}^{1-\kappa_{\max}\sqrt{1-\gamma}}}\big)f(\mathcal{R}^\star)\leq\mathbb{E}[f(\bar{\mathcal{R}})]$$

with the probability of at least $\Big(\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1-2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|\kappa_i}\Big)^T$ *and* $\alpha=1-\Big(2{N^{\mathrm{Ex}}}^2d(\mathcal{G})+\frac{1}{2}{N^{\mathrm{Ex}}}^2+N^{\mathrm{Ex}}\Big)\frac{1}{\lambda}$.

Proof. Let $\bar{\mathbf{x}}(t)=\max\limits_{i\in\mathcal{A}^{\mathrm{Ex}}}\mathbf{x}_i(t)$. It follows from Lemma 1.5.4 and Proposition 4.2.2 that

$$\begin{aligned}&F(\bar{\mathbf{x}}(t+1))-F(\bar{\mathbf{x}}(t))\geq\\&\quad\nabla F(\bar{\mathbf{x}}(t)).(\bar{\mathbf{x}}(t+1)-\bar{\mathbf{x}}(t))-\frac{1}{2}{N^{\mathrm{Ex}}}^2\frac{1}{\lambda^2}f(\mathcal{R}^\star),\end{aligned}$$

which, further from Proposition 4.2.2 we get

$$\begin{aligned}&F(\bar{\mathbf{x}}(t+1))-F(\bar{\mathbf{x}}(t))\geq\\&\quad\frac{1}{\lambda}\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}}\widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t))-\frac{1}{2}{N^{\mathrm{Ex}}}^2\frac{1}{\lambda^2}f(\mathcal{R}^\star).\end{aligned}\tag{5.19}$$

Next, we note that by definition, $\bar{\mathbf{x}}(t)\geq\mathbf{x}_i(t)$ for any $\forall i\in\mathcal{A}^{\mathrm{Ex}}$. Therefore, given Proposition 4.2.2 and Lemma 1.5.7, for any $i\in\mathcal{A}^{\mathrm{Ex}}$, and $p\in\{1,\cdots,n^{\mathrm{Ex}}\}$, we can write

$$\Big|\frac{\partial F}{\partial[\mathbf{x}]_p}(\bar{\mathbf{x}}(t))-\frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_i(t))\Big|\leq N^{\mathrm{Ex}}\frac{1}{\lambda}d(\mathcal{G}))f(\mathcal{R}^\star).\tag{5.20}$$

Knowing that $\mathbf{1}.\widetilde{\mathbf{v}}_i(t)=1$, $i\in\mathcal{A}^{\mathrm{Ex}}$. Consequently, using (5.20) we can write

$$\begin{aligned}&\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}}\widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t))\geq\\&\quad\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}}\widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))-{N^{\mathrm{Ex}}}^2\frac{1}{\lambda}d(\mathcal{G}))f(\mathcal{R}^\star).\end{aligned}\tag{5.21}$$

Next, we let

$$\bar{\mathbf{v}}_i(t) = \underset{\mathbf{w}\in\mathcal{M}_i}{\operatorname{argmax}}\ \mathbf{w}.\nabla F(\bar{\mathbf{x}}(t))$$

and

$$\hat{\mathbf{v}}_i(t) = \underset{\mathbf{w}\in\mathcal{M}_i}{\operatorname{argmax}}\ \mathbf{w}.\nabla F(\mathbf{x}_i(t)).$$

Because f is monotone increasing, we have $\frac{\partial F}{\partial [\mathbf{x}]_p} \geq 0$, we conclude that $\mathbf{1}.\bar{\mathbf{v}}_i(t) = 1$, $i \in \mathcal{A}^{\mathrm{Ex}}$ and also $\mathbf{1}.\hat{\mathbf{v}}_i(t) = 1$, $i \in \mathcal{A}$. Therefore, using $\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))$ and $\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)), i \in \mathcal{A}^{\mathrm{Ex}}$, and (5.20) we can also write

$$\begin{aligned}
&\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \\
&\qquad \sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - {N^{\mathrm{Ex}}}^2 \frac{1}{\lambda} d(\mathcal{G}) f(\mathcal{R}^\star), \qquad (5.22a)\\
&\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)). \qquad (5.22b)
\end{aligned}$$

On the other hand, by virtue of Lemma 1.18, $\frac{\widetilde{\partial F}}{\partial [\mathbf{x}]_p}(\mathbf{x}_i(t))$, $p \in \mathcal{P}_i^{\mathrm{Ex}}$ that each agent $i \in \mathcal{A}^{\mathrm{Ex}}$ uses to solve optimization problem (5.11) satisfies

$$\left| \frac{\widetilde{\partial F}}{\partial [\mathbf{x}]_p}(\mathbf{x}_j(t)) - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_j(t)) \right| \leq \frac{1}{2\lambda} f(\mathcal{R}^\star) \qquad (5.23)$$

with the probability of at least $1 - 2\mathrm{e}^{-\frac{1}{8\lambda^2}K_j}$. Using (5.22b) and (5.23), and also that the samples are drawn independently

$$\begin{aligned}
&\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \\
&\qquad \sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) - N^{\mathrm{Ex}} \frac{1}{2\lambda} f(\mathcal{R}^\star), \qquad (5.24a)\\
&\sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum i \in \mathcal{A}^{\mathrm{Ex}} \hat{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \\
&\qquad \sum\nolimits_{i\in\mathcal{A}^{\mathrm{Ex}}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - N^{\mathrm{Ex}} \frac{1}{2\lambda} f(\mathcal{R}^\star), \qquad (5.24b)
\end{aligned}$$

with the probability of at least $\prod_{i\in\mathcal{A}^{\mathrm{Ex}}} (1 - 2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}$.

From (5.21), (5.22a),(5.24a), and (5.24b) now we can write

$$\sum_{i\in\mathcal{A}^{\mathrm{Ex}}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq$$
$$\sum_{i\in\mathcal{A}^{\mathrm{Ex}}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - (2N^{\mathrm{Ex}}d(\mathcal{G}))+1)N^{\mathrm{Ex}}\frac{1}{\lambda}f(\mathcal{R}^\star), \tag{5.25}$$

with the probability of at least $1-2\sum_{i\in\mathcal{A}^{\mathrm{Ex}}}\mathrm{e}^{-\frac{1}{8\lambda^2}K_i}$.

Next, let $\mathbf{v}_i^\star$ be the projection of $\mathbf{1}_{\mathcal{R}^\star}$ into $\mathcal{M}_i$. Knowing that $\mathcal{M}_i$s are disjoint sub-spaces of $\mathcal{M}$ covering the whole space then we can write $\mathbf{1}_{\mathcal{R}^\star}=\sum_{i\in\mathcal{A}^{\mathrm{Ex}}}\mathbf{v}_i^\star$. Then, using (5.25) and invoking Lemma 1.5.4 and the fact that $\bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t))$ we obtain

$$\sum_{i\in\mathcal{A}^{\mathrm{Ex}}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq$$
$$\sum_{i\in\mathcal{A}^{\mathrm{Ex}}} \mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t)) - (2N^{\mathrm{Ex}}d(\mathcal{G}))+1)N^{\mathrm{Ex}}\frac{1}{\lambda}f(\mathcal{R}^\star) =$$
$$\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\bar{\mathbf{x}}(t)) - (2N^{\mathrm{Ex}}d(\mathcal{G}))+1)N^{\mathrm{Ex}}\frac{1}{\lambda}f(\mathcal{R}^\star) \geq$$
$$f(\mathcal{R}^\star) - F(\bar{\mathbf{x}}(t)) - (2N^{\mathrm{Ex}}d(\mathcal{G}))+1)\frac{N^{\mathrm{Ex}}}{\lambda}f(\mathcal{R}^\star), \tag{5.26}$$

with the probability of at least $\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1-2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}$. Hence, using (5.19) and (5.26), we conclude that

$$F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq$$
$$\frac{1}{\lambda}(f(\mathcal{R}^\star) - F(\bar{\mathbf{x}}(t)) - (2N^{\mathrm{Ex}}d(\mathcal{G})) + \frac{1}{2}N^{\mathrm{Ex}}+1)\frac{N^{\mathrm{Ex}}}{\lambda^2}f(\mathcal{R}^\star), \tag{5.27}$$

with the probability of at least $\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1-2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}$.

Next, let $g(t)=f(\mathcal{R}^\star)-F(\bar{\mathbf{x}}(t))$ and $\beta=(2N^{\mathrm{Ex}}d(\mathcal{G}))+\frac{1}{2}N^{\mathrm{Ex}}+1)\frac{N^{\mathrm{Ex}}}{\lambda^2}f(\mathcal{R}^\star)$, to rewrite (5.27)

as

$$(f(\mathcal{R}^\star) - F(\bar{\mathbf{x}}(t))) - (f(\mathcal{R}^\star) - F(\bar{\mathbf{x}}(t+1))) =$$
$$g(t) - g(t+1) \geq \frac{1}{\lambda}(f(\mathcal{R}^\star) - F(\bar{\mathbf{x}}(t))) - \beta = \frac{1}{\lambda}g(t) - \beta. \tag{5.28}$$

Then from inequality (5.28) we get

$$g(t+1) \leq (1 - \frac{1}{\lambda})g(t) + \beta \tag{5.29}$$

with the probability of at least $\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1 - 2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}$. Solving for inequality (5.29) at time T yields

$$g(T) \leq (1 - \frac{1}{\lambda})^T g(0) + \beta\sum_{k=0}^{T-1}(1 - \frac{1}{\lambda})^k =$$
$$(1 - \frac{1}{\lambda})^T g(0) + \lambda\beta(1 - (1 - \frac{1}{\lambda})^T) \tag{5.30}$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1 - 2\mathrm{e}^{-\frac{1}{8\lambda^2}K_j})^{|\mathcal{P}_i^{\mathrm{Ex}}|}\right)^T$. Substituting back $g(T) = f(\mathcal{R}^\star) - F(\bar{\mathbf{x}}(T))$ and $g(0) = f(\mathcal{R}^\star) - F(\mathbf{x}(0)) = f(\mathcal{R}^\star)$, in (5.30) we then obtain

$$(1 - (1 - \frac{1}{\lambda})^T)(f(\mathcal{R}^\star) - \lambda\beta) =$$
$$(1 - (1 - \frac{1}{\lambda})^T)(1 - (2N^{\mathrm{Ex}}d(\mathcal{G})) + \frac{1}{2}N^{\mathrm{Ex}} + 1)\frac{N^{\mathrm{Ex}}}{\lambda})f(\mathcal{R}^\star)$$
$$\leq F(\bar{\mathbf{x}}(T)), \tag{5.31}$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1 - 2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}\right)^T$. Knowing that $\frac{1}{\mathrm{e}} \geq (1-\frac{1}{\lambda})^\lambda$, we can write $(\frac{1}{\mathrm{e}}) \geq (1-\frac{1}{\lambda})^\lambda$ and $\lambda = \frac{T}{1-\sqrt[\kappa_{\max}]{1-\gamma}}$ from equation (5.9), we can write $(\frac{1}{\mathrm{e}})^{1-\sqrt[\kappa_{\max}]{1-\gamma}} \geq (1 - \frac{1}{\lambda})^T$ and consequently $1 - (\frac{1}{\mathrm{e}})^{1-\sqrt[\kappa_{\max}]{1-\gamma}} \leq 1 - (1 - \frac{1}{\lambda})^T$. Hence, we conclude

$$\alpha(1-\frac{1}{\mathrm{e}^{1-\sqrt[\kappa_{\max}]{1-\gamma}}})f(\mathcal{R}^\star) \leq F(\bar{\mathbf{x}}(T))$$

, with the probability of at least $\left(\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1-2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}\right)^T$ where

$$\alpha = 1-(2N^{\mathrm{Ex}}d(\mathcal{G})+\frac{1}{2}N^{\mathrm{Ex}}+1)\frac{N^{\mathrm{Ex}}}{\lambda}$$

. Moreover, by defining $\bar{\mathbf{z}} = \max_{i\in\mathcal{A}^{\mathrm{Ex}}} \mathbf{z}_i$ and because of equations (5.14), (5.15), we can conclude $[\bar{\mathbf{z}}]_p \geq [\bar{\mathbf{x}}(T)]_p$ for $p \in \mathcal{P}^{\mathrm{Ex}}$. Hence, by the stochastic definition of extended function F given by equation (5.5), we have $F(\bar{\mathbf{x}}) \leq F(\bar{\mathbf{z}})$ and consequently

$$\alpha(1-\frac{1}{\mathrm{e}^{1-\sqrt[\kappa_{\max}]{1-\gamma}}})f(\mathcal{R}^\star) \leq F(\bar{\mathbf{z}}), \tag{5.32}$$

By the stochastic interpretation of the extended function (5.5), we can write

$$F(\bar{\bar{\mathbf{z}}}) = \mathbb{E}[f(\mathcal{R}_{\bar{\mathbf{z}}})]. \tag{5.33}$$

By decomposing $\bar{\mathbf{z}}$ to sub-agent level components, we can write

$$\mathbb{E}[\mathcal{R}_{\bar{\mathbf{z}}}] = \mathbb{E}[f(\bigcup_{a\in\mathcal{A}}\bigcup_{i\in\mathcal{A}_a}\mathcal{R}_{\bar{\mathbf{z}}_i})]. \tag{5.34}$$

Moreover, for a random set $\mathcal{S} \in \mathcal{P}^{\mathrm{Ex}}$ and the random set $\mathcal{R}_{\mathbf{z}_i} = \{p_1, \cdots, p_m\} \subset \mathcal{P}_i^{\mathrm{Ex}},\ i \in \mathcal{A}^{\mathrm{Ex}}$, we have

$$\begin{aligned}
&\mathbb{E}[f(\mathcal{S}\cup\mathcal{R}_{\mathbf{z}_i})] = \mathbb{E}[f(\mathcal{S}) + \sum_{k=1}^{m}\Delta_f(p_k|\mathcal{S}\cup\{p_1,\cdots,p_{l-1}\})] \\
&\leq \mathbb{E}[f(\mathcal{S}) + \sum_{p\in\mathcal{R}_{\mathbf{z}_i}}\Delta_f(p|\mathcal{S})] = \mathbb{E}[f(\mathcal{S}) + \sum_{p\in\mathcal{P}_i^{\mathrm{Ex}}}\Delta_f(p|\mathcal{S})] \\
&= \sum_{p\in\mathcal{P}_i^{\mathrm{Ex}}}[\mathbf{z}_i]_p f(\mathcal{S}\cup\{p\}).
\end{aligned} \tag{5.35}$$

Since $\mathbf{1}^\top.\mathbf{z}_i = 1$ and $[\mathbf{z}_i]_p \geq 0, p \in \mathcal{P}_i^{\mathrm{Ex}}$, then the expression $\sum_{p\in\mathcal{P}_i^{\mathrm{Ex}}}[\mathbf{z}_i]_p f(\mathcal{S}\cup\{p\})$ is equivalent to $\mathbb{E}[f(\mathcal{S}\cup\{p\})]$ when the policy p is chosen randomly according to belief vector

$\mathbf{z}_i$. Hence, by putting equations (5.34) and (5.35), we have

$$\mathbb{E}[\mathcal{R}_{\bar{\mathbf{z}}}] \leq \mathbb{E}[f(\bigcup_{a\in\mathcal{A}}\bigcup_{i\in\mathcal{A}_a} p_i] = \mathbb{E}[f(\bigcup_{a\in\mathcal{A}} \mathcal{T}_a]$$

where p_i is the randomly selected policy according to belief vector $\mathbf{z}_i$ and $\mathcal{T}_a = \bigcup_{i\in\mathcal{A}_a} p_i$ is the set of randomly selected policies of sub-agents of agent a. Having that $f(\mathcal{T}_a) = f(\bar{\mathcal{R}}_a)$ where $\bar{\mathcal{R}}_a = \{\mathsf{PolicyMap}(p) | p \in \mathcal{T}_a\}$, and using the equations (5.32) and (5.33), we can write

$$\alpha(1-\frac{1}{\mathrm{e}^{1-\sqrt[\kappa_{\max}]{1-\gamma}}})f(\mathcal{R}^\star) \leq \mathbb{E}[f(\bigcup_{a\in\mathcal{A}} \mathcal{T}_a] = f(\bar{\mathcal{R}}) \tag{5.36}$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}^{\mathrm{Ex}}}(1-2\mathrm{e}^{-\frac{1}{8\lambda^2}K_i})^{|\mathcal{P}_i^{\mathrm{Ex}}|}\right)^T$, which concludes the proof. □

Theorem 5.2.2 (Privacy characteristics of Algorithm 7)**.** *Given the distributed Algorithm 7 used by each agent $a \in \mathcal{A}$ to solve the policy selection problem 5.1 to achieve the policy set $\bar{\mathcal{R}}_a \in \mathcal{P}_a$ is γ-Private where $\gamma \in [0,1]$ that an intelligent entity other than agent a is only able to estimate $p \in \bar{\mathcal{R}}_a$ with probability of at most γ, for all $p \in \mathcal{P}_a$.*

Proof. Since there are T communication rounds between the agents $i \in \mathcal{A}^{\mathrm{Ex}}$ then due to equations (5.10a) and (5.10a) and given that $\lambda = \frac{T}{1-\sqrt[\kappa_{\max}]{1-\gamma}}$ and the fact that $\mathbf{y}_{ii}$ is generated randomly, then $[\mathbf{z}_i]_p$, $p \in \mathcal{P}^{\mathrm{Ex}}$ can be estimated to be at most $1 - \sqrt[\kappa_{\max}]{1-\gamma}$. Moreover, because the policy set of each sub-agent $i \in \mathcal{A}^{\mathrm{Eq}}$ given by $\mathcal{P}_i^{\mathrm{Ex}} = \{p_i^1, \cdots, p_i^{|\mathcal{P}_a|}\}$ is a copy of policy set of agent a where $i \in \mathcal{A}_a$, and $\mathsf{PolicyMap}(p_i^k) = \mathsf{PolicyMap}(p_j^k)$ where $i, j \in \mathcal{A}_a$ and $k \in \{1, \cdots, |\mathcal{P}_a|\}$, then there are κ_a polices in $\mathcal{P}^{\mathrm{Ex}}$ that are copies of the a single policy in $\mathcal{P}_a$. Since a single policy $p \in \mathcal{P}_i^{\mathrm{Ex}}$ is sampled according to $\mathbf{z}_i$, then each policy in $\mathcal{P}_a$ can be estimated to exist in $\bar{\mathcal{R}}_a$ with probability of at most $1 - (\sqrt[\kappa_{\max}]{1-\gamma})^{\kappa_a} < \gamma$ which concludes the proof. □

The results of Theorem 6.1.2 and Theorem 5.2.2 highlight the trade-off between the size of

Algorithm 7 Distributed γ-Private extension-based algorithm

Init: $\bar{\mathcal{R}} \leftarrow \emptyset,\ \mathbf{x}_i(0) \leftarrow \mathbf{0},\ t \leftarrow 1,$
while $t \leq T$ **do**
  for $i \in \mathcal{A}^{\mathrm{Ex}}$ **do**
    Draw K_i sample policy sets $\mathcal{R}_{\mathbf{x}_i(t)}$.
    for $p \in \mathcal{P}_i^{\mathrm{Ex}}$ **do**
      Calculate $\left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p$ by empirically estimate
    end for
    Solve for $\widetilde{\mathbf{v}}_i(t) = \underset{\mathbf{w}\in\mathcal{M}_i}{\operatorname{argmax}}\, \mathbf{w}.\widetilde{\nabla F}(\mathbf{x}_i(t))$
    Propagate $\mathbf{x}_i^-(t+1) = \mathbf{x}_i(t) + \frac{1}{\lambda}\widetilde{\mathbf{v}}_i(t)$
    Broadcast $\mathbf{x}_i(t)$ to the neighbors $\mathcal{N}_i$.
    Update $\mathbf{x}_i(t+1) = \max_{j\in\mathcal{N}_i\cup\{i\}} \mathbf{x}_j^-(t+1)$
  end for
  $t \leftarrow t+1.$
end while
for $i \in \mathcal{A}^{\mathrm{Ex}}$ **do**
  Generate $\mathbf{y}_{ii}$ and form $\mathbf{z}_i$
end for
for $a \in \mathcal{A}$ **do**
  Sample $\bar{\mathcal{T}}_a$ using $\mathbf{z}_i$, $i \in \mathcal{A}_a$ using Algorithm 8
end for
Map the policies $\bar{\mathcal{R}}_a = \{\mathsf{PolicyMap}(p) | p \in \mathcal{T}_a\}$
Return $\bar{\mathcal{R}} = \bigcup_{a\in\mathcal{A}} \bar{\mathcal{R}}_a$

optimality gap and the guaranteed level of privacy for Algorithm 7. Figure 5.2 shows the trade-off when T is set to a very large value to eliminate the effect of α.

5.3 Conclusion

We proposed a distributed suboptimal algorithm to solve a distributed-constraint problem of maximizing a monotone increasing submodular set function subject to a partition matroid constraint. The main contribution of this chapter was to design a distributed algorithm that enabled each agent to find a suboptimal policy locally with a guaranteed level of privacy. Our algorithm was a distributed solution for the continuous relaxation of the problem of interest followed by a fully distributed loss-less rounding procedure which was done without any inter-agent communication. Instead of using the common differential privacy approach to establish our privacy preservation result, we took advantage of the stochastic nature of

Algorithm 8 Distributed rounding

Input: $a,\ \kappa_a,\ \mathcal{A}_a = \{a_1, \cdots, a_{\kappa_a}\},\ \mathcal{P}_i^{\mathrm{Ex}},\ \mathbf{z}_i,\ i \in \mathcal{A}_a$.
Init: $\mathcal{T}(0) = \emptyset,\ k \leftarrow 1$,
for $k \in \{1, \cdots, \kappa_a\}$ **do**
    Set $i = a_k$
    Generate a random number $\zeta \in [0, 1]$ with a uniform distribution.
    Find $p \in \mathcal{P}_i^{\mathrm{Ex}}$ such that $\sum_{l=1}^{p} [\mathbf{z}_i]_l \leq \zeta \leq \sum_{l=1}^{p+1} [\mathbf{z}_i]_l$
end for
$\mathcal{T}(k) = \mathcal{T}(k-1) \cup \{p\}$
$\mathcal{T}_a = \mathcal{T}(\kappa_a)$
Return $\mathcal{T}_a$

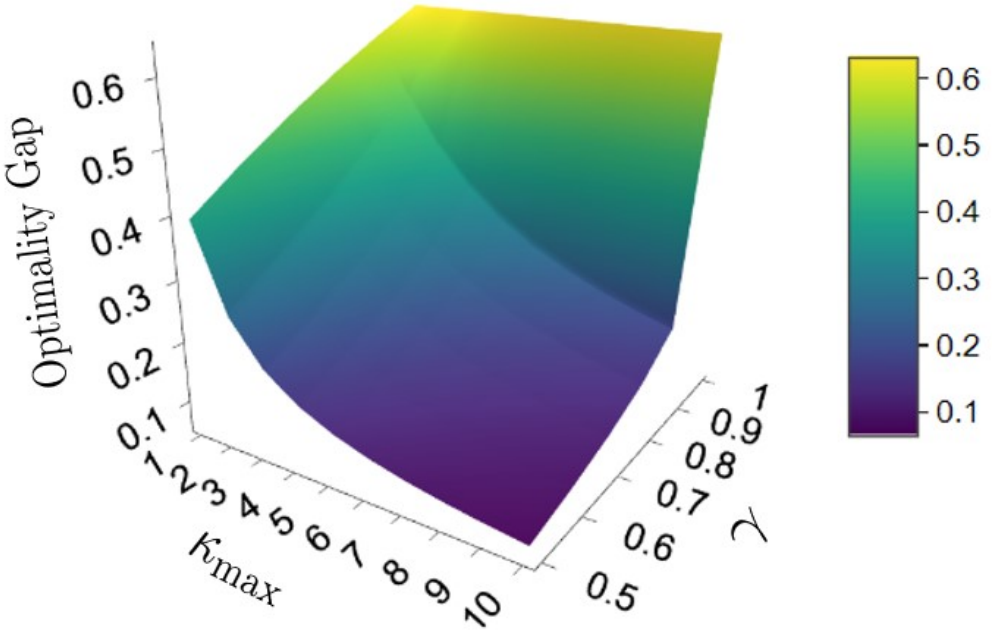


Figure 5.2 – The trade-off between optimality gap and privacy as a function of $\kappa_{\max}$. For $\kappa_{\max} = 1$ and $\gamma = 0.5$ the optimality gap is 0.4 for large values of T.

the rounding procedure. We proposed a novel privacy-preservation framework that tied the level of privacy to a variable $\gamma \in [0, 1]$, which determined the maximum probability that the local policy choice of an agent is revealed. This letter is concluded by formally establishing the trade-off between the worst-case optimality gap and the guaranteed level of privacy.

Chapter 6

Privacy Preservation in Networked Systems

This chapter considers the problem of privacy preservation against passive internal and external eavesdroppers in the continuous-time Laplacian average consensus algorithm over strongly connected and weight-balanced digraphs. For this problem, we evaluate the effectiveness of the use of additive obfuscation signals as a privacy preservation measure against eavesdroppers that know the graph topology. Our results include (a) identifying the necessary and sufficient conditions on admissible additive obfuscation signals that do not perturb the convergence point of the algorithm from the average of initial values of the agents; (b) obtaining the necessary and sufficient condition on the knowledge set of an eavesdropper that enables it to identify the initial value of another agent; (c) designing observers that internal and external eavesdroppers can use to identify the initial conditions of another agent when their knowledge set on that agent enables them to do so. We demonstrate our results through a numerical example.

6.1 Problem Statement

Consider the static average consensus algorithm

$$\dot{x}^i(t) = -\sum_{j=1}^{N} \mathsf{a}_{ij}\,(x^i(t) - x^j(t)), \quad x^i(0) = \mathsf{r}^i, \tag{6.1}$$

over a strongly connected and weight-balanced digraph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{A})$. For such an interaction typology, x^i of each agent $i \in \mathcal{V}$ converges to $\frac{1}{N}\sum_{i=1}^{N} \mathsf{r}$ as $t \to \infty$ [73]. In this algorithm, r^i, represents a *reference value* of agent $i \in \mathcal{V}$. Because in (6.1), the reference value r^i of each agent $i \in \mathcal{V}$ is transmitted to its in-neighbors, this algorithm trivially reveals the reference value r^i of each agent $i \in \mathcal{V}$ to all its in-neighbors and any external agent that is listening to the communication messages. In this chapter, we investigate whether in a network of $N \geq 3$ agents, the reference value of the agents can be concealed from the *eavesdroppers* by adding the obfuscation signals $f^i \in \mathcal{L}_1^\infty$ and $g^i \in \mathcal{L}_1^\infty$ to, respectively, the internal dynamics and the transmitted signal of each agent $i \in \mathcal{V}$ (see Fig. 6.2), i.e.,

$$\dot{x}^i(t) = -\sum_{j=1}^{N} \mathsf{a}_{ij}\,(x^i(t) - y^j(t)) + f^i(t), \tag{6.2a}$$

$$y^i(t) = x^i(t) + g^i(t), \tag{6.2b}$$

$$x^i(0) = \mathsf{r}^i, \tag{6.2c}$$

while still guaranteeing that x^i converges to $\frac{1}{N}\sum_{i=1}^{N} \mathsf{r}$ as $t \to \infty$. We refer to the set of obfuscation signals $\{f^j, g^j\}_{j=1}^N$ in (6.2) for which each agent $i \in \mathcal{V}$ still converges to the average of the reference values across the network, i.e., $\lim_{t\to\infty} x^i(t) = \frac{1}{N}\sum_{i=1}^{N} x(0) = \frac{1}{N}\sum_{i=1}^{N} \mathsf{r}$, as the *admissible obfuscation signals*. We define the eavesdroppers formally as follows.

Definition 3 (eavesdropper)**.** *An eavesdropper is an agent inside (internal agent) or outside (external agent) the network that stores and processes the accessible inter-agent communication messages to obtain the private reference value of the other agents in the network, without*

interfering with the execution of algorithm (6.2).

Definition 4 (Privacy preservation)**.** *Consider an eavesdropper as defined in Definition 3, that has access to* $y^j(t)$, $t \in \mathbb{R}_{\geq 0}$, *of all agents* $j \in \mathcal{O} \subset \mathcal{V}$ *in a network that implements* (6.2) *with locally chosen admissible perturbation signals* (f^l, g^l), $l \in \mathcal{V}$. *We say the privacy of an agent* $i \in \mathcal{V}$ *is preserved if for any arbitrary* $\gamma \in \mathbb{R}_{>0}$, *there exists a tuple* $\{x^{i'}(0) = \mathsf{r}^{i'}, f^{i'}(t), g^{i'}(t)\}$, *with locally chosen admissible perturbations* $(f^{i'}(t), g^{i'}(t))$ *and* $\left|\mathsf{r}^{i'} - \mathsf{r}^{i}\right| > \gamma$, *such that* $y^j(t) \equiv y^{j'}(t)$, $t \in \mathbb{R}_{\geq 0}$, *for all* $j \in \mathcal{O}$.

When privacy of an agent $i \in \mathcal{V}$ is preserved in accordance to Definition 4, it means that there exists arbitrary number of execution of algorithm (6.2) with arbitrary different reference values $\mathsf{r}^{i'}$ ($\left|\mathsf{r}^{i'} - \mathsf{r}^{i}\right| > \gamma$ for any $\gamma \in \mathbb{R}_{>0}$) for agent i for which the signals received by the eavesdropper in all the executions are identical. Privacy preservation according to Definition 4 is stronger than the privacy preservation in stochastic approaches such as [3], where even though the exact reference value is concealed, an estimate with a quantifiable confidence interval on the reference value can be obtained; see Section 6.3 for more discussion.

We examine the privacy preservation properties of algorithm (6.2) against non-collaborative eavesdroppers. The eavesdroppers are non-collaborative if they do not share their *knowledge sets* with each other. The knowledge set of an eavesdropper is the information that it can use to infer the private reference value of the other agents. The extension of our results to collaborative agents is rather straightforward and is omitted for brevity. Without loss of generality, we assume that agent 1 is the internal eavesdropper that wants to obtain the reference value of other agents in the network. At each time $t \in \mathbb{R}_{\geq 0}$, the signals that are available to agent 1 are

$$\mathcal{Y}^1(t) = \{x^1(\tau), y^1(\tau), \{y^i(\tau)\}_{i \in \mathcal{N}^1_{\text{out}}}\}_{\tau=0}^{t}.$$

For an external eavesdropper, the available signals depend on what channels it intercepts.

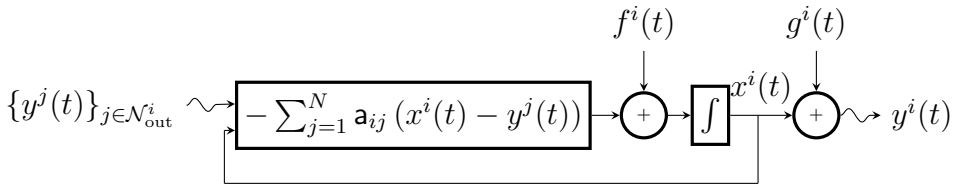


Figure 6.1 – Graphical representation of algorithm 6.2, where f^i and g^i are the additive obfuscation signals.

We assume that the external eavesdropper can associate the intercepted signals to the corresponding agents. We represent the set of these signals with $\mathcal{Y}^{ext}(t)$. We assume that the eavesdropper knows the graph topology. It is also rational to assume that the eavesdroppers are aware of the form of the necessary conditions on the admissible obfuscation signals.

Theorem 6.1.1 (The set of necessary and sufficient conditions on the admissible obfuscation signals)**.** *Consider algorithm* (6.2) *over a strongly connected and weight-balanced digraph with obfuscation signals $f^i, g^i \in \mathcal{L}_1^\infty$, $i \in \mathcal{V}$. Then, the trajectory $t \mapsto x^i(t)$, of all agents $i \in \mathcal{V}$ converges to $\frac{1}{N}\sum_{j=1}^N x^j(0) = \frac{1}{N}\sum_{i=1}^N \mathsf{r}$ as $t \to \infty$ if and only if*

$$\lim_{t\to\infty} \int_0^t \sum_{k=1}^N (f^k(\tau) + \mathsf{d}^k_{\text{out}}\, g^k(\tau))\,\mathrm{d}\tau = 0, \tag{6.3a}$$

$$\lim_{t\to\infty} \int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)}\mathbf{R}^\top(\mathbf{f}(\tau) + \mathbf{A}\,\mathbf{g}(\tau))\,\mathrm{d}\tau = \mathbf{0}, \tag{6.3b}$$

where $\mathbf{L}^+$ and $\mathbf{R}$ are defined in (1.2). □

Proof. To prove necessity, we proceed as follows. We write the algorithm (6.2) in compact form

$$\dot{\mathbf{x}} = -\mathbf{L}\,\mathbf{x} - \mathbf{L}\,\mathbf{g} + \mathbf{f} + \mathbf{D}^{\text{out}}\,\mathbf{g} = -\mathbf{L}\,\mathbf{x} + \mathbf{f} + \mathbf{A}\,\mathbf{g}. \tag{6.4}$$

Left multiplying both sides of (6.4) by $\mathbf{1}_N^\top$ gives

$$\sum\nolimits_{j=1}^{N} \dot{x}^j(t) = \sum\nolimits_{j=1}^{N} (f^i(t) + \mathsf{d}^i_{\text{out}}\, g^i(t)),$$

which results in

$$\sum\nolimits_{j=1}^{N} x^j(t) = \sum\nolimits_{j=1}^{N} x^j(0) + \int_0^t \sum\nolimits_{j=1}^{N} (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\, \mathrm{d}\tau.$$

Because $x^i(0) = \mathsf{r}^i$, to ensure $\lim_{t\to\infty} x^i(t) = \frac{1}{N}\sum_{i=1}^{N} \mathsf{r}$, $i \in \mathcal{V}$, we necessarily need (6.3b).

Next, we apply the change of variable

$$\mathbf{p} = \begin{bmatrix} p_1 \\ \mathbf{p}_{2:N} \end{bmatrix} = \mathbf{T}\,\mathbf{x}, \tag{6.5}$$

where $\mathbf{T}$ is defined in (1.2), to write (6.4) in the equivalent form

$$\dot{p}_1 = \frac{1}{\sqrt{N}} \sum\nolimits_{i=1}^{N} (f^i + \mathsf{d}^i_{\text{out}}\, g^i), \tag{6.6a}$$

$$\dot{\mathbf{p}}_{2:N} = -\mathbf{L}^+\, \mathbf{p}_{2:N} + \mathbf{R}^\top (\mathbf{f} + \mathbf{A}\,\mathbf{g}). \tag{6.6b}$$

The solution of (6.6) is

$$p_1(t) = \frac{1}{\sqrt{N}} \sum\nolimits_{i=1}^{N} x^i(0) + \frac{1}{\sqrt{N}} \int_0^t \sum\nolimits_{i=1}^{N} (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\mathrm{d}\tau, \tag{6.7a}$$

$$\mathbf{p}_{2:N}(t) = \mathrm{e}^{-\mathbf{L}^+ t}\, \mathbf{p}_{2:N}(0) + \int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)} \mathbf{R}^\top (\mathbf{f}(\tau) + \mathbf{A}\,\mathbf{g}(\tau))\, \mathrm{d}\tau. \tag{6.7b}$$

Given (6.3a), (6.7a) results in $\lim_{t\to\infty} p_1(t) = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} x^i(0) = \frac{1}{N}\sum_{i=1}^{N} \mathsf{r}$. Consequently,

given (6.5), to ensure $\lim_{t\to\infty} x^i(t) = \frac{1}{N}\sum_{i=1}^N \mathsf{r}$, $i \in \mathcal{V}$, we need

$$\lim_{t\to\infty} \mathbf{p}_{2:N}(t) = \mathbf{0}. \tag{6.8}$$

Because for a strongly connected and weight-balanced digraph, $-\mathbf{L}^+$ is a Hurwitz matrix, $\lim_{t\to\infty} \mathrm{e}^{-\mathbf{L}^+ t}\mathbf{p}_{2:N}(0) = \mathbf{0}$. Then, the necessary condition for (6.8) is (6.3b).

The sufficiency proof follows from noting that under (6.3), the trajectories of (6.7) satisfy $\lim_{t\to\infty} p_1(t) = \frac{1}{\sqrt{N}}\sum_{i=1}^N x^i(0)$ and $\lim_{t\to\infty} \mathbf{p}_{2:N}(t) = \mathbf{0}$. Then, given (6.5) and $x^i(0) = \mathsf{r}^i$ we obtain $\lim_{t\to\infty} x^i(t) = \frac{1}{N}\sum_{j=1}^N \mathsf{r}^j$, $i \in \mathcal{V}$. □

The necessary and sufficient conditions in (6.3) that specify the admissible signals of the agents are highly coupled. If there exists an ultimately secure and trusted authority that oversees the operation, this authority can assign to each agent its admissible private obfuscation signals that collectively satisfy (6.3). However, in what follows, we consider a scenario where such an authority does not exist, and each agent $i \in \mathcal{V}$, to increase its privacy protection level, wants to choose its own admissible signals (f^i, g^i) privately without revealing them explicitly to the other agents.

Theorem 6.1.2 (Linear algebraic coupling)**.** *Consider algorithm* (6.2) *over a strongly connected and weight-balanced digraph. Let each agent $i \in \mathcal{V}$ choose its local obfuscation signals $f^i, g^i \in \mathcal{L}_1^\infty$ such that*

$$\lim_{t\to\infty} \int_0^t (f^i(\tau) + \mathsf{d}^i_{\mathrm{out}}\, g^i(\tau))\,\mathrm{d}\tau = \beta^i, \tag{6.9}$$

where $\beta^i \in \mathbb{R}$. Then, the necessary and sufficient conditions to satisfy (6.3) *are*

$$\sum\nolimits_{k=1}^N \beta^k = 0, \tag{6.10a}$$

$$\lim_{t\to\infty} \int_0^t \mathrm{e}^{-(t-\tau)} g^i(\tau)\,\mathrm{d}\tau = \alpha \in \mathbb{R}, \quad i \in \mathcal{V}. \tag{6.10b}$$

□

Proof. Given (6.9), it is straightforward to see that (6.10a) is necessary and sufficient for (6.3a). Next, we observe that using (6.9), we can write $\lim_{t\to\infty}\int_0^t \mathbf{R}^\top(\mathbf{f}(\tau)+\mathbf{D}^{\text{out}}\,\mathbf{g}(\tau))\mathrm{d}\tau = \mathbf{R}^\top\begin{bmatrix}\beta^1 & \dots & \beta^N\end{bmatrix}^\top$. Then, it follows from the statement (b) of Lemma 1.5.11 that $\lim_{t\to\infty}\int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)}\mathbf{R}^\top(\mathbf{f}(\tau)+\mathbf{D}^{\text{out}}\,\mathbf{g}(\tau))\mathrm{d}\tau = \mathbf{0}$. As a result, given $\mathbf{f}+\mathbf{A}\,\mathbf{g} = \mathbf{f}+\mathbf{D}^{\text{out}}\,\mathbf{g}-\mathbf{L}\,\mathbf{g}$, we obtain

$$\lim_{t\to\infty}\int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)}\mathbf{R}^\top(\mathbf{f}(\tau)+\mathbf{A}\,\mathbf{g}(\tau))\,\mathrm{d}\tau = \\ -\lim_{t\to\infty}\int_0^t \mathrm{e}^{-\mathbf{L}^+(t-\tau)}\mathbf{R}^\top\mathbf{L}\,\mathbf{g}(\tau)\mathrm{d}\tau. \tag{6.11}$$

Given (6.11), by virtue of Lemma 1.5.10, (6.3b) holds if and only if (6.10b) holds. □

In Theorem 6.1.2, by enforcing condition (6.9) on the admissible signals the coupling between the agents becomes a set of linear algebraic constraints. For a given set of $\{\beta^i\}_{i=1}^N$ and α, Theorem 6.1.2 enables the agents to choose their admissible obfuscation signals locally with guaranteed convergence to the exact average consensus; see Remark 6.1.1 below. Choosing signals that satisfy condition (6.9) is rather easy. However, condition (6.10b) appears to be more complex. The result below, whose proof is given in the appendix, identifies three classes of signals that are guaranteed to satisfy condition (6.10b).

Lemma 6.1.1 (Signals that satisfy (6.10b))**.** *For a given $\alpha\in\mathbb{R}$, let $g = g_1+g_2 \in \mathcal{L}_1^\infty$ satisfy one of the conditions (a) $\lim_{t\to\infty} g(t)=\alpha$ (b) $\lim_{t\to\infty} g_1(t)=\alpha$ and $\lim_{t\to\infty}\int_0^t g_2(\tau)\mathrm{d}\tau = \bar{g}<\infty$ (c) $\lim_{t\to\infty} g_1(t)=\alpha$ and $\int_0^t \sigma(|g_2(\tau)|)\mathrm{d}\tau<\infty$ for $t\in\mathbb{R}_{\geq 0}$, where σ is any class $\mathcal{K}_\infty$ function. Then, $\lim_{t\to\infty}\int_0^t \mathrm{e}^{-(t-\tau)}g(\tau)\mathrm{d}\tau = \alpha$.* □

Proof. When condition (a) holds, the proof of the statement follows from statement (a) of Lemma 1.5.11. When condition (b) is satisfied, the proof follows from the statements (a) and (b) of Lemma 1.5.11 which, respectively, give $\lim_{t\to\infty}\int_0^t \mathrm{e}^{-(t-\tau)}g_1(\tau)\mathrm{d}\tau = \alpha$ and

$\lim_{t\to\infty}\int_0^t \mathrm{e}^{-(t-\tau)}g_2(\tau)\mathrm{d}\tau = 0$. When condition (c) is satisfied, the proof follows from the statement (a) of Lemma 1.5.11 which gives $\lim_{t\to\infty}\int_0^t \mathrm{e}^{-(t-\tau)}g_1(\tau)\mathrm{d}\tau = \alpha$ and noting that $\int_0^t \mathrm{e}^{-(t-\tau)}g_2(\tau)\mathrm{d}\tau$ is the zero state response of system $\dot{\zeta} = -\zeta + g_2$. Since $g_2(t)$ is essentially bounded, this system is ISS, and as a result it is also integral ISS [88]. Then, $\int_0^t \mathrm{e}^{-(t-\tau)}g_2(\tau)\mathrm{d}\tau = 0$, follows from [88, Lemma 3.1]. □

An interesting theoretical finding that Lemma 6.1.1 reveals is that the admissible obfuscation signals $\{f^j, g^j\}_{j=1}^N$, unlike some of the existing results do not necessarily need to be vanishing signals even for $\alpha = 0$ and $\beta^i = 0$, $i \in \mathcal{V}$. For example, $g_1(t) = 0$ and $g_2(t) = \sin(\phi_0 + 2\pi(\frac{c}{2}t^2 + \omega_0 t))$, which is a waveform with linear chirp function [134] where ω_0 is the starting frequency at time $t = 0$, $c \in \mathbb{R}$ is the chirpyness constant, and ϕ_0 is the initial phase, satisfy condition (b) of Lemma 6.1.1 with $\alpha = 0$. This function is smooth but loses its uniform continuity as $t \to \infty$. However, when a non-zero α is used the choices for non-vanishing g that satisfy (6.10b) are much wider, e.g., according to condition (b) of Lemma 6.1.1 any function that asymptotically converges to α can be used.

Remark 6.1.1 (Locally chosen admissible signals)**.** If in a network the agents do not know whether others are going to use obfuscation signals or not, then the agents use $\alpha = 0$ and $\beta^i = 0$, $i \in \mathcal{V}$ in (6.10) and (6.9). This is because the only information available to the agents is that their collective choices should satisfy (6.3). Then, in light of Theorem 6.1.2, to ensure (6.3a) each agent $i \in \mathcal{V}$ chooses its local admissible obfuscation signals according to (6.9) with $\beta^i = 0$. Consequently, according to Theorem 6.1.2 again, each agent $i \in \mathcal{V}$ needs to choose its respective g^i according to (6.10b) with $\alpha = 0$. Any other choice of $\{\beta^i\}_{i=1}^N$ and α needs an inter-agent coordination/agreement procedure. We refer to the admissible signals chosen according to (6.9) and (6.10) as the *locally chosen admissible signals*. □

In the case of the locally chosen admissible obfuscation signals without inter-agent coordination, since the agents need to satisfy (6.9) and (6.10) with $\alpha = \beta^i = 0$, $i \in \mathcal{V}$, these values will

be known to the eavesdroppers. In case that the agents coordinate to choose non-zero values for α and $\{\beta\}_{i=1}^{N}$ such that (6.9) and (6.10) are satisfied, it is likely that these choices to be known to the eavesdroppers. In our privacy preservation analysis below, we consider various cases of the choices of α and/or $\{\beta\}_{i=1}^{N}$ being either known or unknown to the eavesdroppers. This way, our study explains the privacy preservation against the most informed eavesdroppers and also explores what kind of guarantees exists against less informed eavesdroppers that do not know all the parameters. The knowledge sets that we consider are defined as follows.

Definition 5 (Knowledge set of an eavesdropper)**.** The knowledge set of the internal eavesdropper 1 and external eavesdropper ext is assumed to be one of the cases below,

- Case 1:

$$\mathcal{K}^a = \{\mathcal{Y}^a(\infty), \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{A}), \\ \text{form of conditions (6.9) and (6.10)}, \alpha, \{\beta^i\}_{i=1}^{N}\}, \tag{6.12}$$

- Case 2:

$$\mathcal{K}^1 = \{\mathcal{Y}^1(\infty), \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{A}), \\ \text{form of conditions (6.9) and (6.10)}, \alpha\}, \tag{6.13}$$

$$\mathcal{K}^{\text{ext}} = \{\mathcal{Y}^{\text{ext}}(\infty), \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{A}), \\ \text{form of conditions (6.9) and (6.10)}\}, \tag{6.14}$$

- Case 3:

$$\mathcal{K}^{\text{ext}} = \big\{\mathcal{Y}^{\text{ext}}(\infty), \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{A}), \\ \text{form of conditions (6.9) and (6.10)}, \{\beta^i\}_{i=1}^N\big\}, \tag{6.15}$$

where $a \in \{1, \text{ext}\}$. □

Given internal and external eavesdroppers with knowledge sets belonging to one of the cases in Definition 5, our study intends to determine: (a) whether the eavesdroppers inside or outside the network can obtain the reference value of the other agents by storing and processing the transmitted messages; (b) more specifically, what *knowledge set* enables an agent inside or outside the network to discover the reference value of the other agents in the network; (c) what observers such agents can employ to obtain the reference value of the other agents in the network.

6.2 Privacy Preservation Evaluation

In this section, we evaluate the privacy preservation properties of the modified average consensus algorithm (6.2) against an internal eavesdropper 1 and an external eavesdropper whose knowledge sets are either of the two cases given in Definition 5. From the perspective of an eavesdropper interested in private reference value of another agent $i \in \mathcal{V}$, the dynamical system to observe is (6.2) with x^i as the internal state, $(f^i,\ g^i,\ \{y^j\}_{j\in\mathcal{N}^i_{\text{out}}})$ as the inputs and y^i as the measured output. When inputs and measured outputs over some finite time interval (resp. infinite time) are known, the traditional observability (resp. detectability) tests (see [135],[136]) can determine whether the initial conditions of the system can be identified. However, here the inputs f^i and $g^i : \mathbb{R}_{\geq 0} \to \mathbb{R}$ of agent $i \in \mathcal{V}$ are not available to the eavesdropper. All is known is the conditions (6.9) and (6.10) that specify the obfuscation

signals. With regard to inputs $\{y^j\}_{j\in\mathcal{N}^i_{\text{out}}}$ and output y^i, an external agent should intercept these signals while the internal eavesdropper 1 has only access to these inputs if it is an in-neighbor of agent i and all the out-neighbors of agent i (e.g., in Fig. 6.3, agent 1 is an in-neighbor of agent 2 and all the out-neighbors of agent 2).

6.2.1 Case 1 knowledge set

Identifying the initial condition of the agents in the presence of unknown additive obfuscation signals may appear to be related to the classical concept of strong observability/detectability in control theory [137, 138]. However, the necessary conditions on the unknown admissible obfuscation signals provide additional information to the eavesdropper. Such information is not being captured by the strong observability/detectability framework, rendering it inadequate for our study.

It may appear that identifying the initial condition of the agents in the presence of unknown additive obfuscation signals is related to the classical concept of strong observability/detectability in control theory [137, 138]. However, the necessary conditions on the unknown admissible obfuscation signals (6.3) provide additional information to the eavesdropper. Such information is not being captured by the strong observability/detectability framework, rendering it inadequate for our study.

Consider the internal eavesdropper, agent 1, when it intends to obtain the initial condition of one of the agents $i \in \mathcal{V}$. The critical part of the knowledge set of an eavesdropper when it targets an agent is the signals that it has access to. To study privacy preservation for agent $i \in \mathcal{V}$, we partition the graph into islands whose nodes are classified into different groups based on their information exchange by the eavesdropper and its out-neighbors, see Fig. 6.2. For that, note that removing eavesdropper agent 1 and its incident edges results in $\bar{n}^1 \geq 1$ disjoint subgraphs $\bar{\mathcal{G}}^1_k = (\bar{\mathcal{V}}^1_k, \bar{\mathcal{E}}^1_k) \subset \mathcal{G}(\mathcal{V}, \mathcal{E})$, $k \in\in \{1, \cdots, \bar{n}^1\}$. Adding agent 1

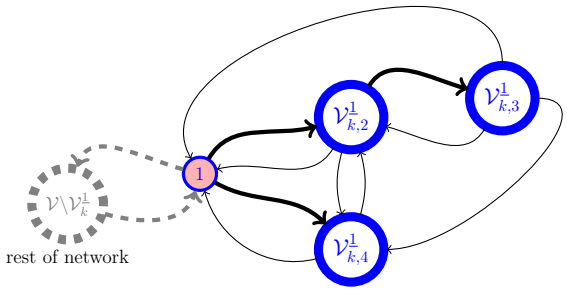


Figure 6.2 – The k^{th} induced island of eavesdropper 1. The super node $\mathcal{V}^1_{k,2}$ in $\mathcal{G}^1_k$ is the set of the out-neighbors of agent 1 that each of them has at least one out-neighbor that is not an out-neighbor of agent 1. The super node $\mathcal{V}^1_{k,4}$ is the set of the out-neighbors of agent 1 whose out-neighbors are all also out-neighbors of agent 1. Finally, the super node $\mathcal{V}^1_{k,3}$ is the set of the agents in $\mathcal{G}^1_k$ that are not an out-neighbor of agent 1. An arrow from each node a (agent 1 or each super node) to another node b (agent 1 or each super node) indicates that at least one agent in a can obtain information from at least one agent in b. The thin connection lines may or may not exist in a network.

in subgraph $\bar{\mathcal{G}}^1_k$ and including its incident edges to this subgraph results in an island graph $\mathcal{G}^1_k = (\mathcal{V}^1_k, \mathcal{E}^1_k) \subset \mathcal{G}(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}^1_k = \bar{\mathcal{V}}^1_k \cup \{1\}$ and $\mathcal{E}^1_k = \{(l,j) \in \mathcal{E} | \, l \in \mathcal{V}^1_k, \ j \in \mathcal{V}^1_k\}$. Every island of agent 1 is connected to the rest of the digraph $\mathcal{G}$ only through agent 1 (see Fig. 6.2). To simplify the notation, with out loss of generality, carry out the subsequent study for agents in island $k = 1$, e.g., $\mathcal{G}^1_1$. Based on how each agent interacts with agent 1, we divide the agents of island $\mathcal{G}^1_1$ into three groups as described below (see Fig. 6.2)

- $\mathcal{V}^1_{1,2} = \big\{ i \in \mathcal{V}^1_1 \big| \, i \in \mathcal{N}^1_{\text{out}}, \ \mathcal{N}^i_{\text{out}} \not\subset \mathcal{N}^1_{\text{out}+1} \big\}$,
- $\mathcal{V}^1_{1,3} = \big\{ i \in \mathcal{V}^1_1 \big| \, i \notin \mathcal{N}^1_{\text{out}} \big\}$.
- $\mathcal{V}^1_{1,4} = \big\{ i \in \mathcal{V}^1_1 \big| \, i \in \mathcal{N}^1_{\text{out}}, \ \mathcal{N}^i_{\text{out}} \subseteq \mathcal{N}^1_{\text{out}+1} \big\}$.

$\mathcal{V}^1_{1,4}$ is the set of the agents that agent 1 has direct access to all their communication signals, while $\mathcal{V}^1_{1,2}$ and $\mathcal{V}^1_{1,3}$ are set of agents that some of inter-agent communication between them is not available to agent 1. Without loss of generality, in what follows we assume that the agents in the network are labeled according to the ordered set $(1, \mathcal{V}^1_{1,2}, \mathcal{V}^1_{1,3}, \mathcal{V}^1_{1,4}, \mathcal{V}\backslash\mathcal{V}^1_1)$. We let the aggregated states and obfuscation signals of the agents in $\mathcal{V}^1_{1,l}$, $l \in \{2,3,4\}$, be

$\mathbf{x}_l = [x^i]_{i\in\mathcal{V}^1_{1,l}}$, $\mathbf{g}_l = [g^i]_{i\in\mathcal{V}^1_{1,l}}$ and $\mathbf{f}_l = [f^i]_{i\in\mathcal{V}^1_{1,l}}$. Similarly, we let the aggregated states and obfuscation signals of the agents in $\mathcal{V}\backslash\mathcal{V}^1_1$ be $\mathbf{x}_5 = [x^i]_{i\in\mathcal{V}\backslash\mathcal{V}^1_1}$, $\mathbf{g}_5 = [g^i]_{i\in\mathcal{V}\backslash\mathcal{V}^1_1}$ and $\mathbf{f}_5 = [f^i]_{i\in\mathcal{V}\backslash\mathcal{V}^1_1}$. We partition $\mathbf{L}$, $\mathbf{A}$ and $\mathbf{D}^{\text{out}}$, respectively, to subblock matrices $\mathbf{L}_{ij}$'s, $\mathbf{A}_{ij}$'s and $\mathbf{D}^{\text{out}}_{ij}$'s in a comparable manner to the partitioned aggregated state $(x^1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5)$(see (**??**)). By definition $\mathbf{L}_{ij} = -\mathbf{A}_{ij}$, $i, j \in \{1, \cdots, 5\}$, $i \neq j$. With the right notation at hand, we present the following result which provides the privacy guarantee according to Definition 4 for the agents belonging to $\mathcal{V}^1_{1,2}$ and $\mathcal{V}^1_{1,3}$. Note that, because every agent in $\mathcal{G}^1_1$ is connected to the rest of the agents in digraph $\mathcal{G}$ only through agent 1, all the out-neighbors and in-neighbors of agent 2 are necessarily in $\mathcal{G}^1_1$.

Lemma 6.2.1 (A case of indistinguishable admissible initial conditions for an internal eavesdropper)**.** *Let agent* 1 *be the internal eavesdropper whose knowledge set is as Definition 6.12. Let* $\mathcal{G}^1_1 = (\mathcal{V}^1_1, \mathcal{E}^1_1)$ *be an island of agent* 1 *that satisfies* $\mathcal{V}^1_{1,2} \neq \{\}$*. Consider the modified static average consensus algorithm* (6.2) *over a strongly connected and weight-balanced digraph* $\mathcal{G}$ *where the agents are implementing* $\{x^i(0) = \mathsf{r}^i, f^i, g^i\}_{i=1}^N$*, with the locally chosen admissible obfuscation signals* (f^i, g^i) *satisfying* (6.9) *and* (6.10)*. Consider also an alternative execution of* (6.2) *with* $\{x^{i'}(0), f^{i'}, g^{i'}\}_{i=1}^N$ *satisfying*

$$
\begin{aligned}
&x^{1'}(0) = x^1(0),\ \mathbf{x}'_4(0) = \mathbf{x}_4(0),\ \mathbf{x}'_5(0) = \mathbf{x}_5(0)\\
&\mathbf{x}'_2(0) - \mathbf{x}_2(0) = -\mathbf{A}_{23}\mathbf{L}_{33}^{-1}(\mathbf{x}'_3(0) - \mathbf{x}_3(0)),
\end{aligned}
\tag{6.16}
$$

and

$$
\begin{aligned}
&f^{i'}(t) = f^i(t), \qquad\qquad i \in \mathcal{V} \setminus \mathcal{V}^1_{1,2}\\
&f^{i'}(t) = f^i(t) - \left[\mathbf{A}_{23}\mathrm{e}^{-\mathbf{L}_{33}t}(\mathbf{x}'_3(0) - \mathbf{x}_3(0))\right]_{i-1},\ \ i \in \mathcal{V}^1_{1,2}
\end{aligned}
\tag{6.17}
$$

and

$$
\begin{aligned}
g^{i'}(t) &= g^{i}(t), && i \in \mathcal{V} \setminus \mathcal{V}^{1}_{1,2} \\
g^{i'}(t) &= g^{i}(t) + \left[\mathrm{e}^{-\mathbf{D}_{22} t} (\mathbf{x}'_2(0) - \mathbf{x}_2(0)), \right]_{i-1}, && i \in \mathcal{V}^{1}_{1,2}
\end{aligned} \tag{6.18}
$$

Then,

$$
y^{i}(t) = y^{i'}(t), \quad t \in \mathbb{R}_{\geq 0}, \qquad i \in \mathcal{V} \backslash \mathcal{V}^{1}_{1,3}. \tag{6.19}
$$

Moreover,

$$
\sum\nolimits_{i=1}^{N} x^{i'}(0) = \sum\nolimits_{i=1}^{N} x^{i}(0) = \sum\nolimits_{i=1}^{N} \mathsf{r}^{i}, \tag{6.20}
$$

$$
\lim_{t \to \infty} x^{i'}(t) = \frac{1}{N} \sum\nolimits_{i=1}^{N} \mathsf{r}^{i}, \qquad i \in \mathcal{V}. \tag{6.21}
$$

□

Proof. If agent 1 knows β^i, the proof follows from Lemma 6.2.2. If agent 1 does not know β^i, since it knows (6.10a), there exists at least one other agent $k \in \mathcal{V} \backslash \{1, i\}$ whose β^k is not known to agent 1. We note that at the best case, $\beta^i + \beta^k$ can be known to agent 1. Now consider $\beta_{ik} \in \mathbb{R} \backslash \{0\}$ and let $\beta^{i'} = \beta^i + \beta_{ik}$ and $\beta^{k'} = \beta^k - \beta_{ik}$, and $\beta^{l'} = \beta^l$ for $l \in \mathcal{V} \backslash \{i, k\}$. Now consider an alternative implementation of algorithm (6.2a)-(6.2b) with initial conditions $x^{l'}(0) = x^l(0)$ for $l \in \mathcal{V} \backslash \{i, k\}$, $x^{i'}(0) = x^i(0) - \beta_{ik}$ and $x^{k'}(0) = x^k(0) + \beta_{ik}$ and obfuscation signals $f^{l'}(t) = f^l(t)$, $g^{l'}(t) = g^l(t)$ for $l \in \mathcal{V} \backslash \{i, k\}$, $f^{i'}(t) = f^i(t) + d\, \beta_{ik} \mathrm{e}^{-(\mathrm{d}^i_{\mathrm{out}} + d)t}$, $g^{i'}(t) = g^i(t) + \beta_{ik} \mathrm{e}^{-(\mathrm{d}^i_{\mathrm{out}} + d)t}$ and $f^{k'}(t) = f^k(t) - d\, \beta_{ik} \mathrm{e}^{-(\mathrm{d}^k_{\mathrm{out}} + d)t}$, $g^{k'}(t) = g^k(t) - \beta_{ik} \mathrm{e}^{-(\mathrm{d}^k_{\mathrm{out}} + d)t}$, where $d \in \mathbb{R}$ is chosen such that $d > \max\{\mathrm{d}^i_{\mathrm{out}}, \mathrm{d}^k_{\mathrm{out}}\}$. Let $t \mapsto x^{l'}(t)$ and $t \mapsto y^{l'}(t)$, $t \in \mathbb{R}_{\geq 0}$, respectively, be the state and the transmitted signal of agent $l \in \mathcal{V}$ in this alternative case. We note that using $\lim_{t \to \infty} \int_0^t d\beta_{ik} \mathrm{e}^{-(\mathrm{d}^i_{\mathrm{out}} + d)\tau} \mathrm{d}\tau = \frac{d\beta_{ik}}{\mathrm{d}^i_{\mathrm{out}} + d}$ and $\lim_{t \to \infty} \int_0^t d\beta_{ik} \mathrm{e}^{-(\mathrm{d}^i_{\mathrm{out}} + d)\tau} \mathrm{d}\tau = \frac{1}{\mathrm{d}^i_{\mathrm{out}} + d}$ we can show $\lim_{t \to \infty} \int_0^t (f^{l'}(\tau) + \mathsf{d}^l_{\mathrm{out}}\, g^{l'}(\tau))\, \mathrm{d}\tau = \beta^{l'}$, and $\lim_{t \to \infty} \int_0^t \mathrm{e}^{-(t-\tau)} g^{l'}(\tau) \mathrm{d}\tau = \alpha$

for $l \in \mathcal{V}$. Therefore, since $\sum_{l=j}^{N} \beta^{j'} = 0$, by virtue of Theorem 6.1.2 we get

$$\lim_{t\to\infty} x^{l'}(t) = \frac{1}{N}\sum\nolimits_{j=1}^{N} x^{l'}(0) = \frac{1}{N}\sum\nolimits_{j=1}^{N} \mathsf{r}^{l}, \quad l \in \mathcal{V}. \tag{6.22}$$

Next, let $\delta x^l(t) = x^l(t) - x^{l'}(t)$ and $\delta y^l(t) = y^l(t) - y^{l'}(t)$, $l \in \mathcal{V}$. Then,

$$\begin{cases} \delta\dot{x}^l(t) = -\mathsf{d}^l_{\text{out}}\delta x^l(t) + \sum\limits_{j=1}^{N} \mathsf{a}_{lj}\delta y^j(t), & l\in\mathcal{V}\backslash\{i,k\}, \\ \delta\dot{x}^l(t) = -\mathsf{d}^l_{\text{out}}\delta x^l(t) + \sum\limits_{j=1}^{N} \mathsf{a}_{lj}\delta y^j(t) + f^l - f^{l'}, & l\in\{i,k\}, \end{cases} \tag{6.23a}$$

$$\begin{cases} \delta y^l(t) = \delta x^l, & l\in\mathcal{V}\backslash\{i,k\}, \\ \delta y^l(t) = \delta x^l + g^l - g^{l'}, & l\in\{i,k\}. \end{cases} \tag{6.23b}$$

To complete our proof, we want to show that $y^l(t) = y^{l'}(t)$ (or equivalently $\delta y^l(t) \equiv 0$), $l \in \mathcal{V}$, for $t \in \mathbb{R}_{\geq 0}$, thus agent 1 cannot distinguish between the initial conditions $x^i(0)$ and $x^{i'}(0)$. Since, for a given initial condition and integrable external inputs the solution of an ordinary differential equation is unique, we achieve this goal by showing that if $\delta y^l(t) = 0$, $l \in \mathcal{V}$ applied in the state dynamics (6.23a), the resulted output (6.23a) satisfy $\delta y^l(t) \equiv 0$, $l \in \mathcal{V}$, $t \in \mathbb{R}_{\geq 0}$. For this, first note that since $\delta x^l(0) = 0$ for $l \in \mathcal{V}\backslash\{i,k\}$, then it follows from (6.23a) with $\delta y^l(t) = 0$, $l \in \mathcal{V}$, that $\delta x^l(t) \equiv 0$. Subsequently, from (6.23b), we get the desired $\delta y^l(t) \equiv 0$, $t \in \mathbb{R}_{\geq 0}$ for $l \in \mathcal{V}\backslash\{i,k\}$. Next, we note that, from (6.23a) with $\delta y^l(t) = 0$, $l \in \mathcal{V}$, given $\delta x^i(0) = \beta_{ik}$ and $\delta x^k(0) = -\beta_{ik}$ we obtain

$$\begin{aligned} \delta x^i(t) =& \beta_{ik}\mathrm{e}^{-\mathsf{d}^i_{\text{out}}t} - \beta_{ik}\mathrm{e}^{-\mathsf{d}^i_{\text{out}}t} + \beta_{ik}\mathrm{e}^{-(\mathsf{d}^i_{\text{out}}+d)t} \\ =& \beta_{ik}\mathrm{e}^{-(\mathsf{d}^i_{\text{out}}+d)t} \\ \delta x^k(t) =& -\beta_{ik}\mathrm{e}^{-\mathsf{d}^k_{\text{out}}t} + \beta_{ik}\mathrm{e}^{-\mathsf{d}^k_{\text{out}}t} - \beta_{ik}\mathrm{e}^{-(\mathsf{d}^k_{\text{out}}+d)t} \\ =& -\beta_{ik}\mathrm{e}^{-(\mathsf{d}^k_{\text{out}}+d)t} \end{aligned}$$

Subsequently, since $g^i - g^{i'} = -\beta_{ik}\mathrm{e}^{-(\mathsf{d}^i_{\text{out}}+d)\tau}$ and $g^k - g^{k'} = \beta_{ik}\mathrm{e}^{-(\mathsf{d}^k_{\text{out}}+d)\tau}$, from (6.23b), we

get the desired $\delta y^l(t) \equiv 0$, $t \in \mathbb{R}_{\geq 0}$ for $l \in \{i, k\}$, which completes our proof. $\square$

Couple of remarks are in order regarding the results of Lemma 6.2.1. First notice that in proof of Lemma 6.2.1, we show that each $(f^{i'}, g^{i'})$, $i \in \mathcal{V}$ satisfies the locally chosen admissible obfuscation signals conditions (6.9) and (6.10) for the same α and β^is used to generate $\{f^i, g^i\}_{i=1}^N$. Next notice that due to (6.16) for any $\gamma \in \mathbb{R}$, there always exists $x^{i'}(0)$ for $i \in (\mathcal{V}^1_{1,2} \cup \mathcal{V}^1_{1,3})$ that satisfies $\left|x^{i'}(0) - x^i(0)\right| > \gamma$, while signals received by the eavesdropper as stated in (6.19), are identical for both execution of the algorithm using $\{x^i(0) = \mathsf{r}^i, f^i, g^i\}_{i=1}^N$ and $\{x^{i'}(0), f^{i'}, g^{i'}\}_{i=1}^N$. This means that the privacy all agents in $(\mathcal{V}^1_{1,2} \cup \mathcal{V}^1_{1,3})$ is preserved in accordance with Definition 4.

We can develop similar results, as stated in the corollary below, for an external eavesdropper that does not have direct access to the output signal of some of the out-neighbors of agent $i \in \mathcal{V}$.

Corollary 6.2.1 (A case of indistinguishable admissible initial conditions for an external eavesdropper)**.** *Let agent* Ext *be the internal eavesdropper whose knowledge set is as Definition 6.12 where the eavesdropper has access to* $y^l(t)$, $l \in \mathcal{O}$ *and agent* k *where the external eavesdropper does not have access to* $y^k(t)$, *i.e.* $k \notin \mathcal{O}$. *Consider the modified static average consensus algorithm* (6.2) *over a strongly connected and weight-balanced digraph* $\mathcal{G}$ *where the agents are implementing* $\{x^i(0) = \mathsf{r}^i, f^i, g^i\}_{i=1}^N$, *with the locally chosen admissible obfuscation signals* (f^i, g^i) *satisfying* (6.9) *and* (6.10). *Consider also an alternative execution of* (6.2) *with* $\{x^{i'}(0), f^{i'}, g^{i'}\}_{i=1}^N$ *satisfying*

$$\begin{aligned} &x^{i'}(0) = x^i(0) \qquad\quad i \in \mathcal{V} \backslash \mathcal{N}^k_{\text{in}} \cup \{k\} \\ &x^{i'}(0) - x^i(0) = -\frac{\mathsf{a}_{ik}}{\mathrm{d}^k_{\text{out}}}(x^{k'}(0) - x^k(0)) \quad i \in \mathcal{N}^k_{\text{in}} \end{aligned} \tag{6.24}$$

and

$$
\begin{aligned}
f^{i'}(t) &= f^i(t) \qquad\qquad i \in \mathcal{V}\backslash\mathcal{N}_{\text{in}}^k \cup \{k\} \\
f^{i'}(t) &= f^i(t) - \mathsf{a}_{ik}\mathrm{e}^{-\mathrm{d}_{\text{out}}^k t}(x^{k'}(0) - x^k(0)) \quad i \in \mathcal{N}_{\text{in}}^k
\end{aligned} \tag{6.25}
$$

and

$$
\begin{aligned}
g^{i'}(t) &= g^i(t) \qquad\qquad i \in \mathcal{V}\backslash\mathcal{N}_{\text{in}}^k \cup \{k\} \\
g^{i'}(t) &= g^i(t) + \mathrm{e}^{-\mathsf{d}_{\text{out}}^i t}(x^{k'}(0) - x^k(0)) \quad i \in \mathcal{N}_{\text{in}}^k
\end{aligned} \tag{6.26}
$$

Then

$$
y^i(t) = y^{i'}(t), \quad t \in \mathbb{R}_{\geq 0}, \qquad\qquad i \in \mathcal{V}\backslash\{k\}. \tag{6.27}
$$

Moreover,

$$
\sum\nolimits_{i=1}^N x^{i'}(0) = \sum\nolimits_{i=1}^N x^i(0) = \sum\nolimits_{i=1}^N \mathsf{r}^i, \tag{6.28}
$$

$$
\lim_{t\to\infty} x^{i'}(t) = \frac{1}{N}\sum\nolimits_{i=1}^N \mathsf{r}^i, \qquad\qquad i \in \mathcal{V}. \tag{6.29}
$$

□

Proof. Proof of Corollary 6.2.1, is straight forward from proof of Lemma 6.2.1. The proof is trivially concluded from equations (6.18),(6.17), and (6.16) through singling out an agent in $\mathcal{V}_{1,3}^1$ and finding all of its in-neighbors in $\mathcal{V}_{1,2}^1$. □

Corollary 6.2.1 shows $(f^{i'}, g^{i'})$, $i \in \mathcal{V}$ satisfies the locally chosen admissible obfuscation signals conditions (6.9) and (6.10) for the same α and β^is used to generate $\{f^i, g^i\}_{i=1}^N$. Next notice that due to (6.24), for any $\gamma \in \mathbb{R}_{>0}$, there always exist $x^{j'}(0)$, $j \in \mathcal{N}_{\text{in}}^k$ and $x^{k'}(0)$ that

satisfies $\left|x^{j'}(0) - x^j(0)\right| > \gamma$ and $\left|x^{k'}(0) - x^k(0)\right| > \gamma$, while the signal transmitted by the agents in $\mathcal{V}\backslash\{k\}$ as stated in (6.27) are identical for both execution of the algorithm using $\{x^i(0) = \mathsf{r}^i, f^i, g^i\}_{i=1}^N$ and $\{x^{i'}(0), f^{i'}, g^{i'}\}_{i=1}^N$. Moreover, since $\mathcal{O} \subset \mathcal{V}\backslash\{k\}$) leads to the fact that the privacy of the agents $\mathcal{N}_{\text{in}}^k \cup \{k\}$ is preserved in accordance with Definition 4.

Through Lemma 6.2.1, we have established that the privacy of agents when the eavesdropper, either internal or external, does not have access to at least one signal that is transmitted in to the agent is preserved. The next results show that such guarantee does not hold for agents whose incoming and outgoing signals are in the knowledge set of the eavesdropper.

Lemma 6.2.2 (Observer design for eavesdroppers with the knowledge set of Case 1)**.** *Consider the modified static average consensus algorithm* (6.2) *with a set of locally chosen admissible obfuscation signals* $\{f^j, g^j\}_{j=1}^N$ *over a strongly connected and weight-balanced digraph* $\mathcal{G}$*. Let the knowledge set of the eavesdroppers be* (6.12)*. An internal eavesdropper agent* 1 *and external eavesdropper* ext *that has access to the output signals of agent* $i \in \mathcal{V}$ *and all its out-neighbors, can employ respectively observer*

$$\dot{\psi} = \sum\nolimits_{j=1}^{N} \mathsf{a}_{ij}(y^i - y^j), \qquad \psi(0) = -\beta^i, \tag{6.30a}$$

$$\nu^1(t) = \psi(t) + x^1(t), \tag{6.30b}$$

and observer

$$\dot{\zeta} = \sum\nolimits_{j=1}^{N} \mathsf{a}_{ij}(y^i - y^j), \quad \zeta(0) = -\beta^i - \alpha, \tag{6.31a}$$

$$\dot{\eta} = -\eta + y^i, \qquad \eta(0) \in \mathbb{R}, \tag{6.31b}$$

$$\nu^{\text{ext}}(t) = \zeta(t) + \eta(t), \tag{6.31c}$$

to asymptotically obtain r^i*,* $i \in \mathcal{V}$*, i.e.,* $\nu^a \to \mathsf{r}^i$*,* $a \in \{\text{ext}, 1\}$ *as* $t \to \infty$*. Moreover, at any*

time $t \in \mathbb{R}_{\geq 0}$, the estimation error of the observers respectively satisfies

$$\nu^1(t) - \mathsf{r}^i = x^1(t) - x^i(t) + \int_0^t (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\mathrm{d}\tau - \beta^i. \tag{6.32}$$

and

$$\nu^{\text{ext}}(t) - \mathsf{r}^i = \eta(t) - x^i(t) + \int_0^t (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\mathrm{d}\tau - \beta^i - \alpha, \tag{6.33a}$$

$$\eta(t) = \mathrm{e}^{-t}\eta_0 + \int_0^t \mathrm{e}^{-(t-\tau)} x^i(\tau)\mathrm{d}\tau + \int_0^t \mathrm{e}^{-(t-\tau)} g^i(\tau)\mathrm{d}\tau. \tag{6.33b}$$

Proof. For an internal eavesdropper, given (6.2) and (6.30) we can write

$$\dot{\psi} + \dot{x}^i = f^i + \mathsf{d}^i_{\text{out}}\, g^i$$

which, because of $x^i(0) = \mathsf{r}^i$ and $\zeta(0) = -\beta^i$, gives

$$\psi(t) = -x^i(t) + \mathsf{r}^i + \int_0^t (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\mathrm{d}\tau - \beta^i,\ t \in \mathbb{R}_{\geq 0}.$$

Then, using (6.30b) and (6.2b) we obtain (6.32) as the estimation error. Subsequently, because of (6.9) and since $\lim_{t\to\infty}(x^1(t) - x^i(t)) = 0$, from (6.32) we obtain $\lim_{t\to\infty} \nu(t) = \mathsf{r}^i$.

For an external eavesdropper, given (6.2) and (6.31a), we can write

$$\dot{\zeta} + \dot{x}^i = f^i + \mathsf{d}^i_{\text{out}}\, g^i,$$

which given $x^i(0) = \mathsf{r}^i$ and $\zeta(0) = -\beta^i - \alpha$, for $t \in \mathbb{R}_{\geq 0}$ gives

$$\zeta(t) = -x^i(t) + \mathsf{r}^i + \int_0^t (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\mathrm{d}\tau - \beta^i - \alpha. \tag{6.34}$$

On the other hand, using (6.2b), $t \mapsto \eta(t)$ is obtained from (6.33b). Then, tracking er-

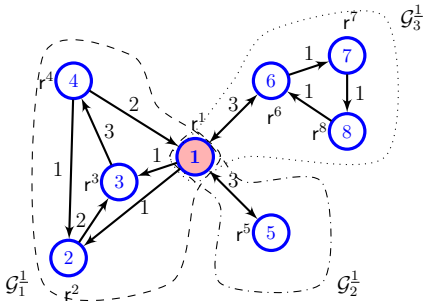


Figure 6.3 – A strongly connected and weight-balanced digraph $\mathcal{G}$ in which node 1 is an articulation point of the undirected representation of $\mathcal{G}$. $\mathcal{G}_1^1$, $\mathcal{G}_2^1$ and $\mathcal{G}_3^1$ are the islands of agent 1.

ror (6.33a) is readily deduced from (6.31c) and (6.34). Next, given (6.9) and (6.10b) and also $\lim_{t\to\infty} \mathrm{e}^{-t}\eta_0 = 0$, we obtain $\lim_{t\to\infty} \nu(t) = \mathsf{r}^i + \lim_{t\to\infty}(-x^i(t) + \int_0^t \mathrm{e}^{-(t-\tau)}x^i(\tau)\mathrm{d}\tau)$.

Subsequently, since $\lim_{t\to\infty} x^i(t) = \frac{1}{N}\sum_{i=1}^N \mathsf{r}$, we can conclude our proof by invoking Lemma 1.5.11 that guarantees $\lim_{t\to\infty} \int_0^t \mathrm{e}^{-(t-\tau)}x^i(\tau)\mathrm{d}\tau = \lim_{t\to\infty} x^i(t) = \frac{1}{N}\sum_{i=1}^N \mathsf{r}$. □

To construct observer (6.30), the internal eavesdropper used its local state. To compensate for the lack of internal state information, the external eavesdropper is forced to employ a higher-order observer (6.31) and invoke condition (6.10b), which the internal eavesdropper does not need. Thus, an external eavesdropper incurs a higher computational cost.

When an eavesdropper does not have direct access to all the signals in $\{y^j(t)\}_{j\in\mathcal{N}_{\text{out}+i}^i}$, a rational strategy appears to be that the eavesdropper estimates the signals it does not have access to. If those agents also have out-neighbors that their output signals are not available to the eavesdropper, then the eavesdropper should estimate the state of those agents as well, until the only inputs to the dynamics that it observes are the additive admissible obfuscation signals. For example, in Fig. 6.3, to obtain the reference value of agent 6, agent 1 compensates for the lack of direct access to $y^7(t)$, which enter the dynamics of agent 6, by estimating the state of all the agents in subgraph $\mathcal{G}_3^1$. Our results below however show that this strategy is not effective. In fact, we show that an eavesdropper (internal or external) is able to uniquely identify the reference value of an agent $i \in \mathcal{V}$ if and only if it has direct

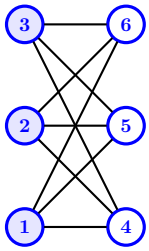
access to $\{y^j(t)\}_{j\in\mathcal{N}^i_{\text{out}+i}}$ for all $t \in \mathbb{R}_{\geq 0}$.

Building on our results so far, we are now ready to state the necessary and sufficient condition under which an eavesdropper with knowledge set (6.12) can discover the reference value of an agent $i \in \mathcal{V}$.

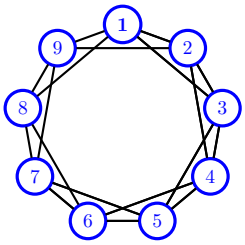
Theorem 6.2.1 (Privacy preservation using the modified average consensus algorithm (6.2) when the knowledge set of the eavesdroppers is given by Case 1 in Definition 5)**.** *Consider the modified static average consensus algorithm* (6.2) *with a set of locally chosen admissible obfuscation signals* $\{f^i, g^i\}_{i=1}^N$ *over a strongly connected and weight-balanced digraph* $\mathcal{G}$*. Let the knowledge set of the internal eavesdropper* 1 *and external agent* ext *be* (6.12)*. Then, (a) agent* 1 *can reconstruct the exact initial value of agent* $i \in \mathcal{V}\backslash\{1\}$ *if and only if* $i \in \mathcal{N}^1_{\text{out}}$ *and* $\mathcal{N}^i_{\text{out}} \subseteq \mathcal{N}^1_{\text{out}+1}$*; (b) the external agent* ext *can reconstruct the exact initial value of agent* $i \in \mathcal{V}$ *if and only if* $\{\{y^j(\tau)\}_{j\in\mathcal{N}^i_{\text{out}+i}}\}_{\tau=0}^{\infty} \subseteq \mathcal{Y}^{\text{ext}}(\infty)$*.*

Proof. Proof of statement (a): If $i \in \mathcal{N}^1_{\text{out}}$ and $\mathcal{N}^i_{\text{out}} \subseteq \mathcal{N}^1_{\text{out}+1}$, Lemma (6.2.2) guarantees that agent 1 can employ an observer to obtain the reference value of agent i. Next, we show that if $i \notin \mathcal{N}^1_{\text{out}}$ or $\mathcal{N}^i_{\text{out}} \not\subset \mathcal{N}^1_{\text{out}+1}$, then agent 1 cannot uniquely identify the reference value r^i of agent i. Suppose agent $i \in \mathcal{V}\backslash\{1\}$ satisfies $i \notin \mathcal{N}^1_{\text{out}}$ (resp. $i \in \mathcal{N}^1_{\text{out}}$ and $\mathcal{N}^i_{\text{out}} \not\subset \mathcal{N}^1_{\text{out}+1}$). Without loss of generality let $\mathcal{V}^1_1$ be the island of agent 1 that contains this agent i. Consequently, $i \in \mathcal{V}^1_{1,3}$ (resp. $i \in \mathcal{V}^1_{1,2}$). Then, by virtue of Lemma 6.2.1, we know that there exists infinite number of alternative admissible initial conditions and corresponding admissible obfuscation signals for any agents in $\mathcal{V}^1_{1,3} \cup \mathcal{V}^1_{1,2}$ for which the time histories of each signal transmitted to agent 1 are identical. Therefore, agent 1 cannot uniquely identify the initial condition of any agents in $\mathcal{V}^1_{1,3} \cup \mathcal{V}^1_{1,2}$. In light of Lemma 6.2.2 and Corollary 6.2.1, the proof of statement (b) is similar to that of statement (a) and is omitted for brevity. □

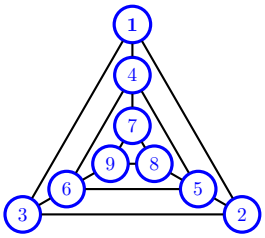
Remark 6.2.1 (Privacy preserving graph topologies)**.** There are several classes of undirected



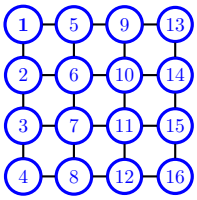
(a) A cyclic bipartite undirected connected graph.



(b) A 4-regular ring lattice undirected connected graph on 9 vertices.



(c) A triangular stacked prism graph.



(d) A lattice graph with 16 vertices (a biconnected graph that contains no cycle with 3 edges).

Figure 6.4 – Examples of privacy-preserving graph topologies.

graphs for which any two agents on the graph have an exclusive neighbor with respect to the other. Thus, by Theorem 6.2.1 privacy of all the agents is preserved from any internal eavesdropper when they implement algorithm (6.2). Examples include cyclic bipartite undirected graphs, 4-regular ring lattice undirected graphs with $N > 5$, planar stacked prism graphs, directed ring graphs, and any biconnected undirected graph that does not contain a cycle with 3 edges (see [139] for the formal definition of these graph topologies). Some examples of these privacy-preserving topologies are shown in Fig. 6.4. Theorem 6.2.1 also presents an opportunity to make agents private with respect to a particular or all the other agents by rewiring the graph so that the conditions of the theorem are satisfied. The idea of rewiring the graph to induce privacy preservation has been explored in the literature [140, 141, 142, 143]. However, in practice, rewiring may be infeasible or costly. □

Next, we show that even though agent 1 cannot obtain the initial condition of the individual

agents in $\mathcal{V}^1_{k,2} \neq \{\}$ and $\mathcal{V}^1_{k,3}$, $k \in \{1, \cdots \bar{n}^1\}$, it can obtain the average of the initial conditions of those agents. Without loss of generality, we demonstrate our results for $k = 1$.

Proposition 6.2.2 (Island anonymity)**.** *Consider the dynamic consensus algorithm* (6.2) *over a strongly connected and weight-balanced digraph $\mathcal{G}$ in which $\mathcal{V}^1_{1,2} \neq \{\}$. Let $n_{2,3} = |\mathcal{V}^1_{1,2} \cup \mathcal{V}^1_{1,3}|$ and $\mathsf{d}^{1,1}_{\text{out}} = \sum_{j \in (\mathcal{V}^1_{1,2} \cup \mathcal{V}^1_{1,4})} \mathsf{a}_{1j}$ be the out-degree of agent 1 in subgraph $\mathcal{G}^1_1$. Then, the eavesdropper 1 with the knowledge set* (6.12) *can employ the observer*

$$
\begin{aligned}
\dot{\zeta}_i &= \sum\nolimits_{j=1}^{N} \mathsf{a}_{ij}(y^i - y^j), \qquad \zeta_i(0) = -\beta^i, \quad i \in \mathcal{V}^1_{1,4}, \\
\dot{\eta} &= -\sum_{j \in (\mathcal{V}^1_{1,2} \cup \mathcal{V}^1_{1,4})} a_{1j}(y^1 - y^j), \qquad \eta(0) = -\sum\nolimits_{j \in \mathcal{V}^1_1 \setminus \{1\}} \beta^i, \\
\mu(t) &= \frac{\eta(t) - \sum_{i \in \mathcal{V}^1_{1,4}} \zeta_i}{n_{2,3}} + x^1(t).
\end{aligned}
$$

to have $\lim_{t \to \infty} \mu(t) = \frac{1}{n_{2,3}} \sum_{j \in (\mathcal{V}^1_{1,2} \cup \mathcal{V}^1_{1,3})} \mathsf{r}^j$.

Proof. Consider the aggregate dynamics of η and $\mathbf{x}_i$, $i \in \{2, 3, 4\}$, which reads as

$$
\begin{bmatrix} \dot{\eta} \\ \dot{\mathbf{x}}_2 \\ \dot{\mathbf{x}}_3 \\ \dot{\mathbf{x}}_4 \end{bmatrix} = - \underbrace{\begin{bmatrix} \mathsf{d}^{1,1}_{\text{out}} & -\mathbf{A}_{12} & \mathbf{0} & -\mathbf{A}_{14} \\ -\mathbf{A}_{21} & \mathbf{D}^{\text{out}}_{22} & -\mathbf{A}_{23} & -\mathbf{A}_{24} \\ -\mathbf{A}_{31} & -\mathbf{A}_{32} & \mathbf{D}^{\text{out}}_{33} & \mathbf{0} \\ -\mathbf{A}_{41} & -\mathbf{A}_{42} & \mathbf{0} & \mathbf{D}^{\text{out}}_{44} \end{bmatrix}}_{\mathsf{L}^1_1} \begin{bmatrix} y^1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \\ \mathbf{y}_4 \end{bmatrix} + \\
\begin{bmatrix} 0 \\ \mathbf{f}_2 + \mathbf{D}^{\text{out}}_{22}\mathbf{g}_2 \\ \mathbf{f}_3 + \mathbf{D}^{\text{out}}_{33}\mathbf{g}_3 \\ \mathbf{f}_4 + \mathbf{D}^{\text{out}}_{44}\mathbf{g}_4 \end{bmatrix}.
$$

Notice that $\mathbf{L}^1_1$ is the Laplacian matrix of graph $\mathcal{G}^1_1$. By Virtue of Lemma 1.5.12 in the appendix we know that $\mathcal{G}^1_1$ is a strongly connected and weight-balanced digraph. Consequently,

left multiplying both sides of equation above with $\mathbf{1}^\top_{|\mathcal{V}_1^1|}$ gives

$$\dot{\eta} + \sum_{j\in\mathcal{V}_1^1\setminus\{1\}} x^i = \sum_{j\in\mathcal{V}_1^1\setminus\{1\}} (f^j(t) + \mathsf{d}^j_{\text{out}}\, g^j(t)).$$

Thereby, given $\eta(0) = -\sum_{j\in\mathcal{V}_1^1\setminus\{1\}} \beta^i$ and $x^i(0) = \mathsf{r}^i$, we obtain

$$\eta(t) = \sum_{j\in\mathcal{V}_1^1\setminus\{1\}} \mathsf{r}^j - \sum_{j\in\mathcal{V}_1^1\setminus\{1\}} x^j(t) + \sum_{j\in\mathcal{V}_1^1\setminus\{1\}} \int_0^t (f^j(\tau) + \mathsf{d}^j_{\text{out}}\, g^j(\tau))\mathrm{d}\tau - \sum_{j\in\mathcal{V}_1^1\setminus\{1\}} \beta^i.$$

On the other hand, following the proof of Lemma 6.2.2, we can conclude that

$$\sum_{i\in\mathcal{V}_{1,4}^1} \zeta_i(t) = \sum_{i\in\mathcal{V}_{1,4}^1} \mathsf{r}^i - \sum_{i\in\mathcal{V}_{1,4}^1} x^i(t) + \sum_{i\in\mathcal{V}_{1,4}^1} \int_0^t (f^i(\tau) + \mathsf{d}^i_{\text{out}}\, g^i(\tau))\mathrm{d}\tau - \sum_{i\in\mathcal{V}_{1,4}^1} \beta^i.$$

Therefore, we can write

$$n_{2,3}\,\mu(t) = \sum_{j\in(\mathcal{V}_{1,2}^1\cup\mathcal{V}_{1,3}^1)} \mathsf{r}^i - \sum_{j\in(\mathcal{V}_{1,2}^1\cup\mathcal{V}_{1,3}^1)} x^i(t) - \sum_{j\in(\mathcal{V}_{1,2}^1\cup\mathcal{V}_{1,3}^1)} \beta^i + \sum_{j\in(\mathcal{V}_{1,2}^1\cup\mathcal{V}_{1,3}^1)} \int_0^t (f^j(\tau) + \mathsf{d}^j_{\text{out}}\, g^j(\tau))\mathrm{d}\tau + n_{2,3}\, x^1(t).$$

The proof then follows from the necessary condition (6.9) on the obfuscation signals, and the fact that $\lim_{t\to\infty} n_{2,3}\, x^1(t) - \sum_{j\in(\mathcal{V}_{1,2}^1\cup\mathcal{V}_{1,3}^1)} x^i(t) = 0$ (recall that $\lim_{t\to\infty} x^i(t) = \lim_{t\to\infty} x^j(t)$, $\forall i, j \in \mathcal{V}$). □

6.2.2 Case 2 and Case 3 knowledge sets

The first result below shows that if β^i corresponding to the locally chosen admissible obfuscation signals of an agent $i \in \mathcal{V}$ is not known to the eavesdropper, the privacy of the agent i is preserved even if the eavesdropper knows all the transmitted input and output signals of agent i and the parameter α. The proof of this lemma is given in the appendix.

Lemma 6.2.3 (Privacy preservation for $i \in \mathcal{V}$ via a concealed β^i)**.** *Consider the modified static average consensus algorithm* (6.2) *with a set of locally chosen admissible obfuscation signals* $\{f^j, g^j\}_{j=1}^N$ *over a strongly connected and weight-balanced digraph* $\mathcal{G}$*. Let the knowledge set of the eavesdropper* 1 *include the form of conditions* (6.9) *and* (6.10)*, and also the parameter* α *that the agents agreed to use. Let agent* 1 *be the in-neighbor of agent* $i \in \mathcal{V}$ *and all the out-neighbors of agent* i*, i.e., agent* 1 *knows* $\{y^j(t)\}_{j\in\mathcal{N}^i_{\text{out}+i}}$, $t \in \mathbb{R}_{\geq 0}$*. Then, the eavesdropper* 1 *can obtain* r^i *of agent* i *if and only if it knows* β^i*.*

A similar statement to that of Lemma 6.2.3 can be made about an external eavesdropper. In the case of the external eavesdropper, it is very likely that the eavesdropper does not know α, as well. Building on the result of Lemma 6.2.3, we make our final formal privacy preservation statement as follows.

Theorem 6.2.3 (Privacy preservation using the modified average consensus algorithm (6.2) when the knowledge set of the eavesdroppers is given by Case 2 in Definition 5)**.** *Consider the modified static average consensus algorithm* (6.2) *with a set of locally chosen admissible obfuscation signals* $\{f^j, g^j\}_{j=1}^N$ *over a strongly connected and weight-balanced digraph* $\mathcal{G}$*. Let the knowledge set of the internal eavesdropper* 1 *and the external eavesdropper* ext *be given by Case 2 in Definition 5. Then, the eavesdropper 1 (resp. agent* ext*) cannot reconstruct the reference value* r^i *of any agent* $i \in \mathcal{V}\backslash\{1\}$ *(resp.* $i \in \mathcal{V}$*).*

Proof. Any agent $i \in \mathcal{V}\backslash\{1\}$ satisfies either $\mathcal{N}^i_{\text{out}+i} \subset \mathcal{N}^1_{\text{out}+1}$ or $\mathcal{N}^i_{\text{out}+i} \not\subset \mathcal{N}^1_{\text{out}+1}$. Since the

eavesdropper 1 does not know $\{\beta^i\}_{j=2}^N$, if $\mathcal{N}^i_{\text{out}+i} \subset \mathcal{N}^1_{\text{out}+1}$, $i \in \mathcal{V}\backslash\{1\}$, (agent 1 has access to all the transmitted input and output signals of agent i), it follows from Lemma 6.2.3 that it cannot reconstruct r^i. Consequently, if $\mathcal{N}^i_{\text{out}+i} \not\subset \mathcal{N}^1_{\text{out}+1}$, $i \in \mathcal{V}\backslash\{1\}$, since the eavesdropper 1 lacks more information (it does not have access to some or all of the transmitted input and output signals of agent i), we conclude that the eavesdropper 1 cannot reconstruct r^i. The proof of the statement for the external eavesdropper is similar to that of the internal eavesdropper 1, and is omitted for brevity (note here that the external eavesdropper ext lacks the knowledge of α, as well). □

Next we show that in fact, knowing β^i, $i \in \mathcal{V}$, e.g., when it is known that agents use $\beta^i = 0$, does not result in the breach of privacy against external eavesdroppers that do not know α.

Theorem 6.2.4 (Privacy preservation using the modified average consensus algorithm (6.2) when the knowledge set of the eavesdroppers is given by Case 3 in Definition 5)**.** *Consider the modified static average consensus algorithm* (6.2) *with a set of locally chosen admissible obfuscation signals* $\{f^j, g^j\}_{j=1}^N$ *over a strongly connected and weight-balanced digraph* $\mathcal{G}$*. Let the knowledge set of the external eavesdropper* ext *be given by Case 3 in Definition 5. Then, the eavesdropper* ext *cannot reconstruct the reference value* r^i *of any agent* $i \in \mathcal{V}$*.*

Proof. The transmitted out signals of the agents implementing (6.2) with the locally chosen admissible obfuscation signals $\{f^j, g^j\}_{j=1}^N$ are $\mathbf{y}(t) = \mathrm{e}^{-\mathcal{L}t}\mathbf{x}(0) + \int_0^t \mathrm{e}^{-\mathbf{L}(t-\tau)}(\mathbf{f}(\tau) + \mathbf{A}\,\mathbf{g}(\tau))\,\mathrm{d}\tau + \mathbf{g}(t)$. Now consider an alternative implementation of (6.2) with initial conditions $\mathbf{x}'(0) = \mathbf{x}(0) - a\mathbf{1}$ and $\mathbf{g}'(t) = \mathbf{g}(t) + a\mathbf{1}$ and $\mathbf{f}'(t) = \mathbf{f}(t) - [\mathsf{d}^1_{\text{out}}, \cdots, \mathsf{d}^N_{\text{out}}]^\top a$ for any $a \in \mathbb{R}$. Note that $\{f^{j'}, g^{j'}\}_{j=1}^N$ are valid locally chosen admissible obfuscation signals that satisfy (6.9) and (6.10a) with the same parameter β^i, $i \in \mathcal{V}$ of $\{f^j, g^j\}_{j=1}^N$ and satisfy (6.10b) with $\alpha + a$ where α is the parameter of (6.10b) corresponding to $\{g^j\}_{j=1}^N$. The transmitted out signal of the agents in this implementation are $\mathbf{y}'(t) = \mathrm{e}^{-\mathcal{L}t}\mathbf{x}'(0) + \int_0^t \mathrm{e}^{-\mathbf{L}(t-\tau)}(\mathbf{f}'(\tau) + \mathbf{A}\,\mathbf{g}'(\tau))\,\mathrm{d}\tau + \mathbf{g}'(t) = \mathrm{e}^{-\mathcal{L}t}\mathbf{x}(0) - a\mathrm{e}^{-\mathcal{L}T}\mathbf{1} + \int_0^t \mathrm{e}^{-\mathbf{L}(t-\tau)}(\mathbf{f}(\tau) + \mathbf{A}\,\mathbf{g}(\tau) - [\mathsf{d}^1_{\text{out}}, \cdots, \mathsf{d}^N_{\text{out}}]^\top a + \mathbf{A}\mathbf{1}a)\,\mathrm{d}\tau + \mathbf{g}'(t) = \mathbf{x}(t) -$

$a\mathrm{e}^{-\mathcal{L}t}\mathbf{1} + \mathbf{g}(t) + a\mathbf{1} = \mathbf{x}(t) + \mathbf{g} = \mathbf{y}(t)$, where we used $\mathrm{e}^{-\mathcal{L}t}\mathbf{1} = \mathbf{1}$. Since the eavesdropper ext does not know the parameter of the condition (6.10b) and $\mathbf{y}'(t) \equiv \mathbf{y}(t)$ for all $t \in \mathbb{R}_{\geq 0}$, it cannot distinguish between the actual and the alternative implementations. Therefore, it cannot uniquely identify the initial condition of the agents. □

Remark 6.2.2 (Guaranteed privacy preservation when an ultimately secure authority assigns the admissible obfuscation signals)**.** If there exists an ultimately secure and trusted authority that assigns the agents' admissible private obfuscation signals in a way that they collectively satisfy (6.3), the privacy of the agents is not trivially guaranteed. This is because it is rational to assume that the eavesdroppers know the necessary condition (6.3) and may be able to exploit it to their benefit. However, in light of Theorem 6.2.4, we are now confident to offer the privacy preservation guarantee for such a case. This is because in this case, the eavesdroppers' knowledge set lacks more information than Case 2 in Definition 5 (note that the locally chosen admissible obfuscation signals are a specially structured subset of all the possible classes of the admissible obfuscation signals).

6.3 Performance Demonstration

6.3.1 Stochastic vs. deterministic privacy preservation

The deterministic and stochastic approaches to privacy preservation withhold different definitions of a private agent. In our deterministic setup, privacy is preserved when an eavesdropper, despite its knowledge set, ends up in an underdetermined system of equations when it wants to obtain the reference value of an agent. Therefore, the eavesdropper is left with infinite guesses of a private agent's reference value, which it cannot favor any of them more than the other. However, the stochastic privacy of an agent is preserved when the eavesdropper's estimate of the reference value yields a non-zero uncertainty. For example, in [3]

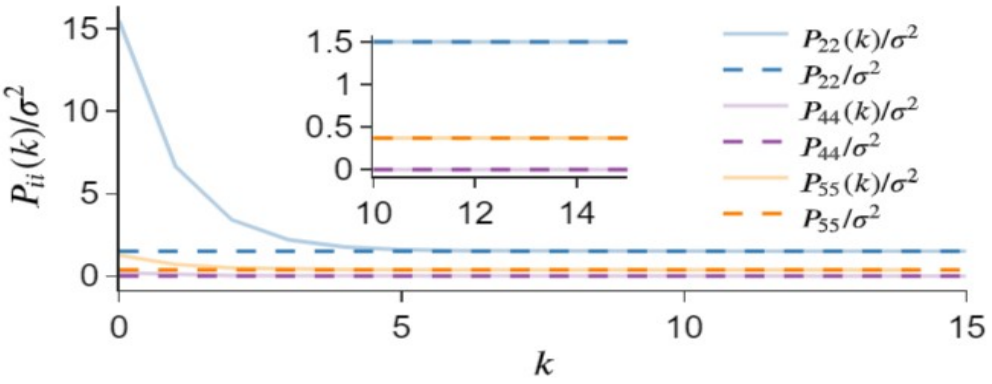


Figure 6.5 – Agent 1's (eavesdropper) maximum likelihood estimator's result when method of [3] is used over graph of Fig. 6.6(a).

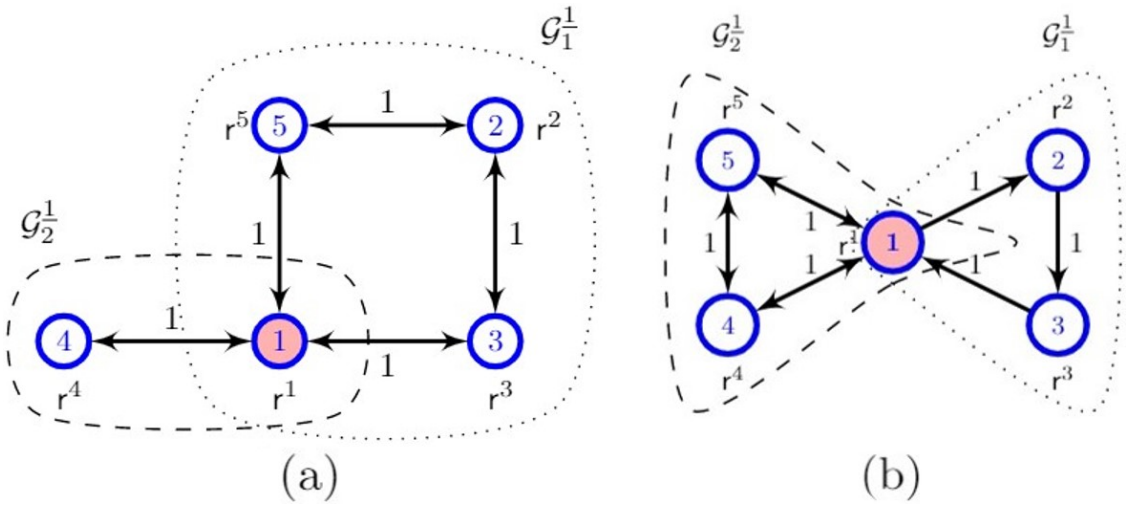


Figure 6.6 – Two strongly connected and weight-balanced graphs $\mathcal{G}$.

a maximum likelihood estimator is used by the eavesdropper to estimate the reference value of the other agents. It is shown that the variance of $P(k)$ of this estimator converges to a constant matrix P. The privacy statement determines that the agents' privacy whose corresponding component in P converges to zero is breached. More specifically, given a vector ζ, a space of the agents' initial condition $\zeta^\top \mathbf{x}(0)$ is disclosed to the eavesdropper if $\zeta^\top P\zeta = 0$ and if $\zeta^\top P\zeta > 0$, it is interpreted as conserving the privacy of the subspace. In this setting, for an agent whose corresponding component of P is non-zero, the eavesdropper does not know the agent's exact reference value, but it has an estimate on it. Hence, we tend to favor the deterministic notion of privacy over stochastic as the deterministic approach reveals less information. Figure 6.5 is the replicate of the result of an example study over the graph in Fig. 6.6(a) in [3], which shows the evolution of the covariance of the maximum-likelihood estimator of the eavesdropper. As expected P_{44} converges to zero but P_{22} and P_{55} not. Even though P_{22} and P_{55} are non-zero, they are pretty small, indicating that the eavesdropper can have a good estimate of the reference values of these agents. In contrast in our work, our privacy preservation shows that for agents whose privacy is preserved, the eavesdropper not only cannot obtain the reference value but also cannot establish an estimate.

Consider the network given in Fig 6.6(a). To demonstrate over results consider the following three implementations of the modified continuous-time Laplacian average consensus

algorithm (6.2) with the reference values and the additive obfuscation signals as follows:

$$
\begin{aligned}
&\text{Case (1)}: \quad \mathbf{r} = [-3, 5, 1, -2, 10]^\top, \\
&\mathbf{f}(t) = [-3, -2\cos(\frac{t}{t^2+1}), \frac{t^5}{t^5+1}, \tan(\frac{\pi}{4}\tanh(t)), -2\tanh(t)]^\top, \\
&\mathbf{g}(t) = [1 + 0.23\mathrm{e}^{-t}, \cos(10\pi\frac{t^2}{t^5+1}), (1 + \mathrm{e}^{-t}\sin(10t))\tanh(t), \\
&\qquad 1 + \mathrm{e}^{(t-1)^2}, \log(\mathrm{e} - \mathrm{e}^{0.1t}(1 + \sin(t)))]^\top.
\end{aligned}
$$

$$
\begin{aligned}
&\text{Case (2)}: \quad \mathbf{r} = [-3, 15, -4, -2, 5]^\top, \\
&\mathbf{f}(t) = [-3, -2\cos(\frac{t}{t^2+1}), -10\mathrm{e}^{-2t} + \frac{t^5}{t^5+1}, \tan(\frac{\pi}{4}\tanh(t)), \\
&\qquad -10\mathrm{e}^{-2t} - 2\tanh(t)]^\top, \\
&\mathbf{g}(t) = [1 + 0.23\mathrm{e}^{-t}, \cos(10\pi\frac{t^2}{t^5+1}), \\
&\qquad 5\mathrm{e}^{-2t} + (1 + \mathrm{e}^{-t}\sin(10t))\tanh(t), 1 + \mathrm{e}^{(t-1)^2}, \\
&\qquad 5\mathrm{e}^{-2t} + \log(\mathrm{e} - \mathrm{e}^{0.1t}(1 + \sin(t)))]^\top.
\end{aligned}
$$

$$
\begin{aligned}
&\text{Case (3)}: \quad \mathbf{r} = [-3, 25, -9, -2, 0]^\top, \\
&\mathbf{f}(t) = [-3, -2\cos(\frac{t}{t^2+1}), -20\mathrm{e}^{-2t} + \frac{t^5}{t^5+1}, \tan(\frac{\pi}{4}\tanh(t)), \\
&\qquad -20\mathrm{e}^{-2t} - 2\tanh(t)]^\top, \\
&\mathbf{g}(t) = [1 + 0.23\mathrm{e}^{-t}, \cos(10\pi\frac{t^2}{t^5+1}), \\
&\qquad 10\mathrm{e}^{-2t} + (1 + \mathrm{e}^{-t}\sin(10t))\tanh(t), 1 + \mathrm{e}^{(t-1)^2}, \\
&\qquad 10\mathrm{e}^{-2t} + \log(\mathrm{e} - \mathrm{e}^{0.1t}(1 + \sin(t)))]^\top.
\end{aligned}
$$

Let Case (1) correspond to the actual operation case, and the other two cases be admissible alternative ones. Here, all the admissible obfuscation signals are smooth, uniformly continuous and non-vanishing. They satisfy (6.9), (6.10a) and (6.10b) with $\alpha = 1$ and $\beta^i = 0$, $i \in \mathcal{V} = \{1, 2, 3, 4, 5\}$. The plots in the top row of Fig. (6.10) confirms convergence of the algorithm to the exact average, as guaranteed by Theorem 6.2. The plots in the sec-

ond row of Fig. (6.10) show that the transmitted-out signal y^i of each agent $i \in \mathcal{V}$ satisfies $\lim_{t\to\infty} y^i(t) = \frac{1}{N}\sum_{j=1}^{N} \mathsf{r}^j + \alpha$. Let $\delta y^i(t)$ be the communication signals difference between Case (j), $j \in \{2, 3\}$ and Case (1). As seen in the two bottom plots in Fig 6.10, only $\delta y^2(t)$ is non-zero. This means that agent 1, in all three cases, receives exactly the same transmission messages from its neighbors, agents 4, 5, and 3. This result, as predicted by Theorem 6.2, shows that agent 1, the eavesdropper, cannot tell whether r^2 is equal to 5 of Case (1), 15 of Case (2) or 25 Case (3). Moreover, agent 1 is not able to say which one of these cases is more probable. A similar statement can be made about agent 3 and 5 whose privacy is guaranteed in our framework. While the privacy of agent 2, 3 and 5 is preserved, according to Lemma 6.2.2, agent 1 can employ an observer of form (6.30) to asymptotically estimate the reference value of agent 4. The response of this estimator is shown in Fig. 6.11. Here to make a comparison study with respect to [3], we used the undirected graph of Fig 6.6(a).

6.3.2 Performance over a digraph with external and internal eavesdroppers

The first demonstration study we conduct is using execution of the modified static average consensus algorithm (6.2) over the strongly connected and weight-balanced digraph in Fig. 6.6(b). The reference value and the locally chosen obfuscation signals of the agents are

$$\begin{aligned}
&\mathsf{r}^1 = 3,\ \mathsf{r}^2 = 2,\ \mathsf{r}^3 = 5, \mathsf{r}^4 = -3,\ \mathsf{r}^5 = -1,\\
&f^l(t) = \mathsf{d}^l_{\text{out}}(\sin(l\frac{\pi}{12}) + \cos(l\frac{\pi}{12}))\frac{\sqrt{(2\,l)}}{4l}\mathrm{e}^{-t}, \qquad (6.35)\\
&g^l(t) = \sin(l\frac{\pi}{12} + l\pi t^2), \qquad\qquad l \in \mathcal{V}.
\end{aligned}$$

The locally chosen admissible obfuscation signals here satisfy the conditions in Theorem 6.2.1 with $\alpha = 0$ and $\beta^i = 0$, $i \in \mathcal{V}$. The interested reader can examine these conditions con-

veniently using the online integral calculator [144]. Let the eavesdropper be agent 1 whose knowledge set is (6.12) (Case 1 in Definition 5). With regards to agents 4 and 5, despite use of non-vanishing obfuscation signals g^4 and g^5, as guaranteed in Lemma 6.2.2, agent 1 can employ local observers of the form (6.30) to obtain $x^4(0) = \mathsf{r}^4 = -3$ and $x^5(0) = \mathsf{r}^5 = -1$ (see Fig. 6.8). Agent 1 however, cannot uniquely identify r^2 and r^3, since $\mathcal{N}^2_{\text{out}} = \{3\} \not\subseteq \mathcal{N}^1_{\text{out}+1} = \{1, 2, 4, 5\}$. To show this, consider an *alternative* implementation of algorithm (6.2) with initial conditions and admissible obfuscation signals

$$
\begin{aligned}
&x^{1\prime}(0)=3,\ x^{2\prime}(0)=1,\ x^{3\prime}(0)=6,\ x^{4\prime}(0)=-3,\ x^{5\prime}(0)=-1,\\
&f^{i\prime}(t)=f^i(t), \qquad\qquad g^{i\prime}(t)=g^i(t), \qquad i\in\{1,3,4,5\},\\
&f^{2\prime}(t)=f^2(t)-\mathrm{e}^{-t}, \quad g^{2\prime}(t)=g^2(t)+\mathrm{e}^{-t},
\end{aligned}
\tag{6.36}
$$

where $\frac{1}{5}\sum_{i=1}^{5} x^{i\prime}(0) = \frac{1}{5}\sum_{i=1}^{5} x^i(0) = \frac{1}{5}\sum_{i=1}^{8} \mathsf{r}^i = 1.2$. As Fig. 6.7 shows the execution of algorithm (6.2) using the initial conditions and obfuscation signals (6.35) (the actual case) and those in (6.36) (an alternative case) converge to the same final value of 1.2. Let $\delta y^i = y^i - y^{i\prime}$, $i \in\in \{1, \cdots, 5\}$ be the error between the output of the agents in the actual and the alternative cases. As Fig. 6.7 shows $\delta y^i \equiv 0$ for all $i \in \mathcal{N}^1_{\text{out}} = \{2, 4, 5\}$. This means that agent 1 cannot distinguish between the actual and the alternative cases and therefore, fails to identify uniquely the initial values of agent 2 and also agent 3. Figure 6.9 shows that an external eavesdropper that has access to the output signals of agents 2 and its knowledge set is (6.12) can employ an observer of the form (6.31) to identify the initial value of agent 2, i.e., $\mathsf{r}^2 = 2$.

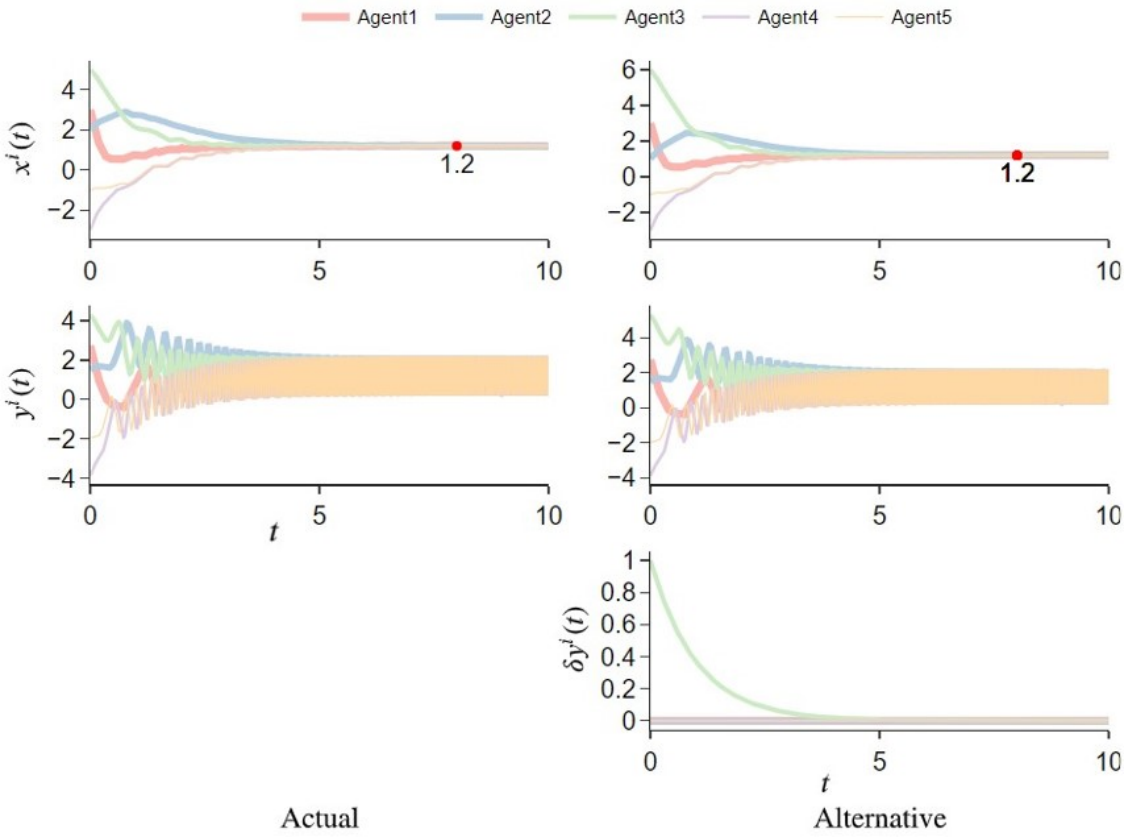


Figure 6.7 – Trajectories of the state of the agents under the actual initial conditions and the obfuscation signals (6.35) as well as the alternative ones in (6.36) and time history of the difference between the output signal of an agent in actual implementation scenario and its output signal in the alternative implementation described in (6.36).

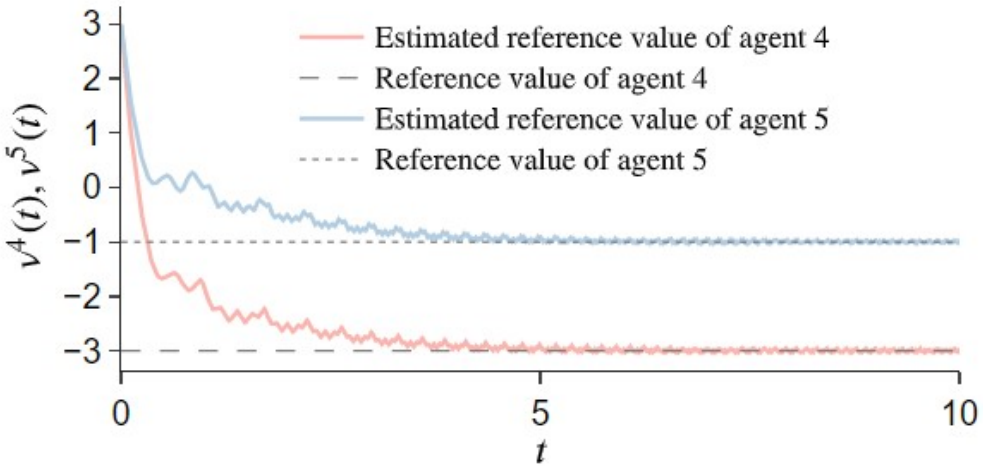


Figure 6.8 – Time history of the observers of the form (6.30) that agent 1 with knowledge set (6.12) uses to obtain r^4 and r^5.

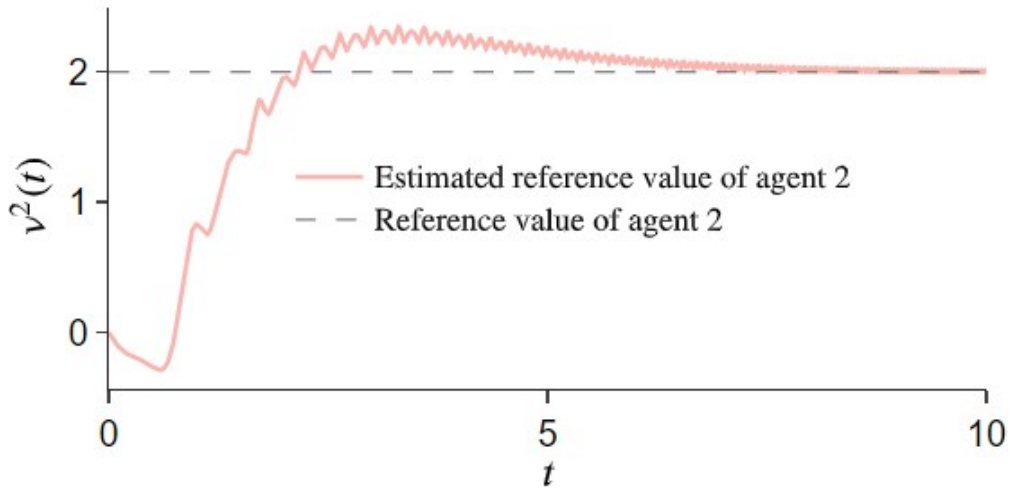


Figure 6.9 – Time history of the observer (6.31) of an external eavesdropper with knowledge set (6.12) that wants to obtain r^2 and has direct access to y^2 and y^3 for all $t \in \mathbb{R}_{\geq 0}$.

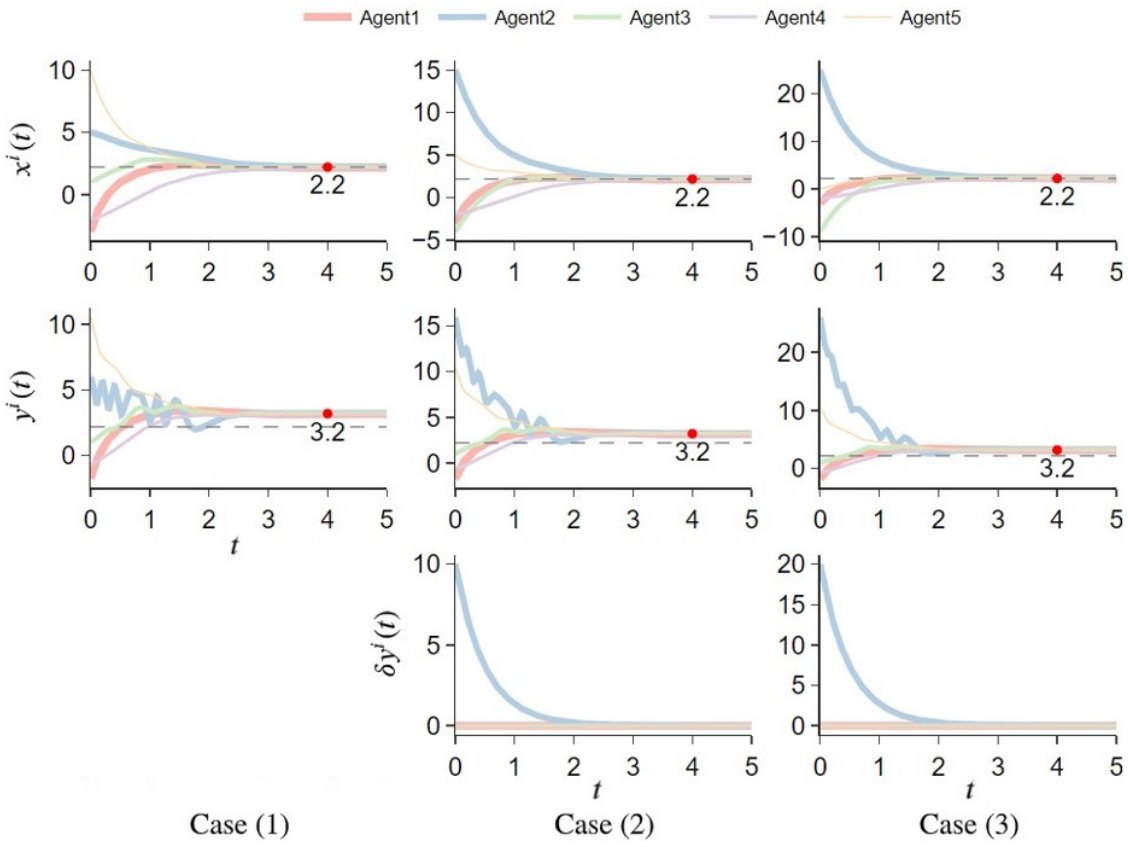


Figure 6.10 – The consensus results for 3 different cases.

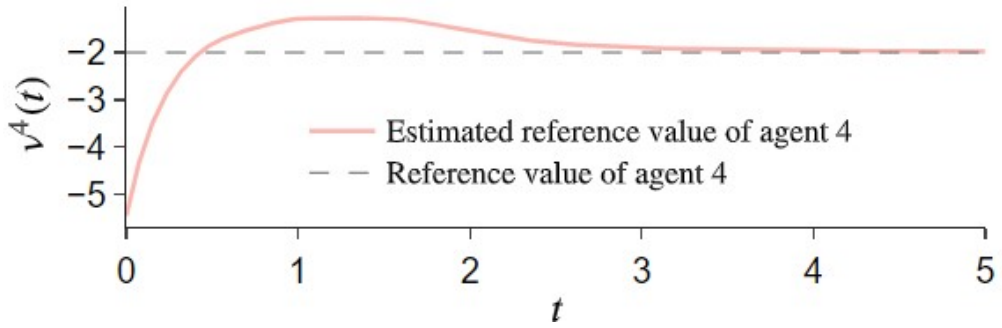


Figure 6.11 – Privacy breach of agent 4 in all 3 cases of 6.6(a).

6.4 Conclusions

In this chapter, we considered the problem of preserving the privacy of the reference value of the agents in an average consensus algorithm using additive obfuscation signals. We started our study by characterizing the set of the necessary and sufficient conditions on the admissible obfuscation signals, which do not perturb the final convergence point of the algorithm. We assessed the privacy preservation property of the average consensus algorithm with the additive obfuscation signals against internal and external eavesdroppers, depending on how much knowledge the eavesdroppers have about the necessary conditions that specify the class of signals that the agents choose their local admissible obfuscation signals from. We showed that if the necessary conditions are fully known to the eavesdroppers, then an internal or external eavesdropper that has access to all the transmitted input and out signals of an agent can employ an asymptotic observer to obtain the reference value of that agent. Next, we showed that indeed having access to all the transmitted input and out signals of an agent at all $t \in \mathbb{R}_{\geq 0}$ is the necessary and sufficient condition for an eavesdropper to identify the initial value of that particular agent. On the other hand, we showed that if the necessary conditions defining the locally chosen admissible obfuscation signals are not fully known to the eavesdroppers, then the eavesdroppers cannot reconstruct the reference value of any other agent in the network. Our future work includes extending our results to other multi-agent distributed algorithms such as dynamic average consensus and distributed

optimization algorithms.

Chapter 7

Certified Neural Networks

This chapter proposes a data-driven method for learning convergent control policies from offline data using Contraction theory. Contraction theory enables constructing a policy that makes the closed-loop system trajectories inherently convergent towards a unique trajectory. At the technical level, identifying the contraction metric, which is the distance metric with respect to which a robot's trajectories exhibit contraction is often non-trivial. We propose to jointly learn the control policy and its corresponding contraction metric while enforcing contraction. To achieve this, we learn an implicit dynamics model of the robotic system from an offline data set consisting of the robot's state and input trajectories. Using this learned dynamics model, we propose a data augmentation algorithm for learning contraction policies. We randomly generate samples in the state-space and propagate them forward in time through the learned dynamics model to generate auxiliary sample trajectories. We then learn both the control policy and the contraction metric such that the distance between the trajectories from the offline data set and our generated auxiliary sample trajectories decreases over time. We evaluate the performance of our proposed framework on simulated robotic goal-reaching tasks and demonstrate that enforcing contraction results in faster convergence and greater robustness of the learned policy.

7.1 Problem Statement

We consider the problem of control policy design for a robot with unknown discrete-time dynamics model $f(\mathbf{x}, \mathbf{u}) : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$, where $\mathcal{X} \in \mathbb{R}^n$ is convex, $\mathcal{U} \in \mathbb{R}^m$. We assume that we can use an offline data set $\mathcal{D}$ consisting of tuples of state transitions and control inputs $(\mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{u}_t)$ satisfying unknown system dynamics

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t). \tag{7.1}$$

Our objective is to obtain a data-driven state-feedback control policy $\mathbf{u}_t = \mathbf{u}(\mathbf{x}_t)$ to steer the system (7.1) towards a desired reference state $\mathbf{x}^r \in \mathbb{R}^n$, i.e. $\mathbf{x}_t \to \mathbf{x}^r$ as $t \to \infty$. To compensate for the lack of knowledge of the true system dynamics, we propose using a model of the system dynamics that we learn from the offline data $\mathcal{D}$. Note that this indicates that our learned dynamics model may still not be available explicitly and may only be available as implicit dynamics such as neural networks approximators. More specifically, we aim to design a control policy $\mathbf{u}(\mathbf{x}_t)$ that leverages the learned dynamics model

$$\mathbf{x}^{'}_{t+1} = f^{'}(\mathbf{x}_t, \mathbf{u}_t), \tag{7.2}$$

to drive the system asymptotically to $\mathbf{x}^r$.

7.2 Learning Deep Contraction Policies

To develop a policy that results in contractive behavior, we seek to enforce the approximate condition in (1.37), requiring the weighted distances of close neighboring trajectories to decrease over time. To enforce this condition, we need to ensure that we have sufficiently close neighboring points for each point within our training set. However, our training data

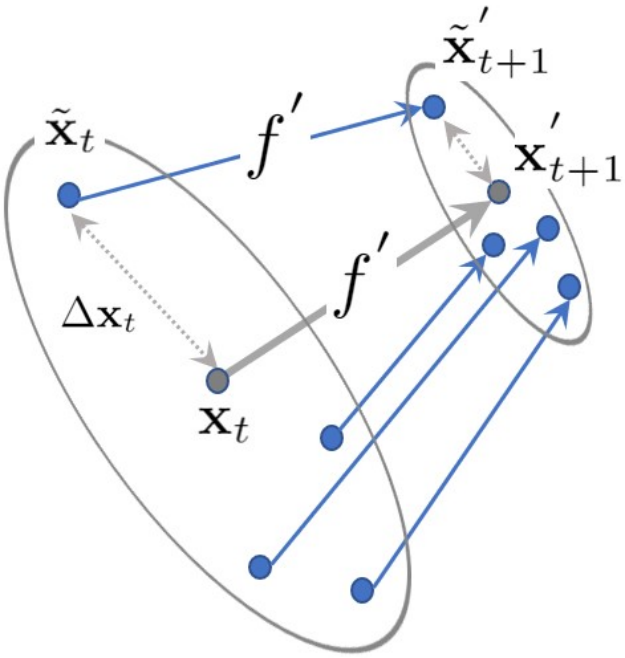


Figure 7.1 – We sample a small displacement $\Delta\mathbf{x}_t$ around the data point $\mathbf{x}_t$ to augment an auxiliary point $\tilde{\mathbf{x}}_t = \mathbf{x}_t + \Delta\mathbf{x}_t$ to our data set. Then, we propagate the auxiliary state $\tilde{\mathbf{x}}_t$ and the actual state $\mathbf{x}_t$ through our learned dynamics model $f^{'}$ under feedback control law $\mathbf{u}(\mathbf{x}_t)$ to calculate the next states: $\tilde{\mathbf{x}}^{'}_{t+1}$ and $\mathbf{x}^{'}_{t+1}$, respectively. Finally, we require that the weighted distance between the two states decreases over time as stated in condition (7.4).

set may not include such neighboring trajectories. We augment our data set with auxiliary trajectories that enable us to enforce this condition at each data point. That is, for each $\mathbf{x}_t \in \mathcal{D}$, we augment our data set with a $\Delta\mathbf{x}_t$ sampled from

$$\Delta_t = \left\{ \Delta\mathbf{x}_t \in \mathbb{R}^n \,\middle|\, \|\Delta\mathbf{x}_t\| < \lambda \right\}, \tag{7.3}$$

where the parameter λ is set in the training process. We sample points from Δ_t to ensure that $\Delta\mathbf{x}_t$ is a small displacement with respect to the training data set. Then, for each data point $\mathbf{x}_t$, we create the auxiliary state $\tilde{\mathbf{x}}_t = \mathbf{x}_t + \Delta\mathbf{x}_t$. Both of these points are propagated through our learned dynamics model to calculate the states at the next time step: $\mathbf{x}_{t+1}^{'} = f^{'}(\mathbf{x}_t, \mathbf{u}(\mathbf{x}_t))$ and $\tilde{\mathbf{x}}_{t+1}^{'} = f^{'}(\tilde{\mathbf{x}}_t, \mathbf{u}(\tilde{\mathbf{x}}_t))$. The initial state, the auxiliary state, and the predicted evolution of these two states are then combined into a tuple $(\mathbf{x}_t, \tilde{\mathbf{x}}_t, \mathbf{x}_{t+1}^{'}, \tilde{\mathbf{x}}_{t+1}^{'})$. The collection of all such tuples over each $\mathbf{x}_t \in \mathcal{D}$ form the augmented data set $\mathcal{D}'$.

Now, we want to evaluate the contracting behavior of the controller $\mathbf{u}(\mathbf{x}_t)$ through the learned model on the augmented data set $\mathcal{D}'$. Thus, we seek to enforce condition (1.37) for the elements of $\mathcal{D}'$

$$\|\boldsymbol{\Theta}(\mathbf{x}_{t+1}^{'})(\tilde{\mathbf{x}}_{t+1}^{'} - \mathbf{x}_{t+1}^{'})\| - \|\boldsymbol{\Theta}(\mathbf{x}_t)\,\Delta\mathbf{x}_t\| < 0, \tag{7.4}$$

with respect to a contraction metric $\boldsymbol{\Theta}(\mathbf{x}_t)$. Contractive behavior is illustrated in Figure 7.1, showing the weighted distance between $\tilde{\mathbf{x}}_t$ and $\mathbf{x}_t$ decays as the system evolves to $\tilde{\mathbf{x}}_{t+1}^{'}$ and $\mathbf{x}_{t+1}^{'}$. We evaluate the approximate contraction condition only at the states $\mathbf{x}_t$ that exist in the data set $\mathcal{D}$. This is due to the fact that the dynamics model is learned from $\mathcal{D}$ and hence $f^{'}$ is expected to behave the most accurately at these points, which in turn will increase the quality of the learned policy. This will enforce contractive behavior with respect to the learned dynamics model f'. Later we will discuss how we can ensure contractive behavior of the closed-loop behavior of the true dynamics model f.

Since in general, the contraction metric $\boldsymbol{\Theta}(\mathbf{x})$ is not known, and it is directly coupled to the control policy, we propose a learning-based approach to jointly learn both the control policy and the metric with respect to which the policy exhibits contraction. We refer to such a policy as a deep contraction policy. To this end, let us start by assuming that we know a control policy $\mathbf{u}(\mathbf{x})$ that makes $f^{'}(\mathbf{x}, \mathbf{u}(\mathbf{x}))$ contractive. Consider now that we want to learn a corresponding contraction metric. Let this contraction metric be represented by a model $\hat{\boldsymbol{\Theta}}(\mathbf{x}; \mathbf{w}_\Theta)$ which is parameterized by weights $\mathbf{w}_\Theta$. We then obtain the best parameters of this contraction metric, denoted by $\mathbf{w}_\Theta^\star$, from

$$\mathbf{w}_\Theta^\star = \underset{\mathbf{w}_\Theta}{\operatorname{argmin}} \; L_{\boldsymbol{\Theta}}(\mathcal{D}'; \mathbf{w}_\Theta), \tag{7.5}$$

where

$$L_{\boldsymbol{\Theta}}(\mathcal{D}'; \mathbf{w}_\Theta) = \mathbb{E}_{\mathcal{D}'}\Big(\|\hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1}; \mathbf{w}_\Theta)(\tilde{\mathbf{x}}^{'}_{t+1} - \mathbf{x}^{'}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t; \mathbf{w}_\Theta)\Delta\mathbf{x}_t\| \Big).$$

The term $\|\hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1}; \mathbf{w}_\Theta)(\tilde{\mathbf{x}}^{'}_{t+1} - \mathbf{x}^{'}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t; \mathbf{w}_\Theta)\Delta\mathbf{x}_t\|$ is an approximate measure of the contraction condition (7.4) which ideally should be negative for all elements of $\mathcal{D}'$. Since enforcing (7.4) directly results in a non-differentiable optimization, we minimize (7.6) as a proxy for (7.4). Note that $L_{\boldsymbol{\Theta}}$ is computed over all data points in $\mathcal{D}'$. When paired with differentiable contraction metric $\hat{\boldsymbol{\Theta}}(\mathbf{x}; \mathbf{w}_\Theta)$, the choice of loss function (7.6) is differentiable and is amenable to gradient decent optimization.

Now, let's consider the more general case where both the policy and its contraction metric are unknown. We want to learn both the state-feedback policy and the contraction metric together. We want to learn a control policy represented by a function approximator $\hat{\mathbf{u}}(\mathbf{x}; \mathbf{w}_\mathbf{u})$, parameterized by weights $\mathbf{w}_\mathbf{u}$, such that the closed-loop system is contractive with respect to the metric model $\hat{\boldsymbol{\Theta}}$. To achieve this, we propagate the initial data points in $\mathcal{D}'$ with the control policy model $\hat{\mathbf{u}}(\mathbf{x}; \mathbf{w}_\mathbf{u})$ as $\mathbf{x}^{'}_{t+1} = f^{'}(\mathbf{x}_t, \hat{\mathbf{u}}(\mathbf{x}_t; \mathbf{w}_\mathbf{u}))$ and $\tilde{\mathbf{x}}^{'}_{t+1} = f^{'}(\tilde{\mathbf{x}}_t, \hat{\mathbf{u}}(\tilde{\mathbf{x}}_t; \mathbf{w}_\mathbf{u}))$.

We obtain the parameters of the contraction metric $\mathbf{w}_\Theta$, denoted by $\mathbf{w}_\Theta^\star$, and the parameters of the control policy $\mathbf{w}_\mathbf{u}$, denoted by $\mathbf{w}_\mathbf{u}^\star$, by minimizing a loss function L_u over the data set $\mathcal{D}'$

$$(\mathbf{w}_\mathbf{u}^\star, \mathbf{w}_\Theta^\star) = \underset{\mathbf{w}_\mathbf{u}, \mathbf{w}_\Theta}{\operatorname{argmin}} \; L_u(\mathcal{D}'; \mathbf{w}_\mathbf{u}, \mathbf{w}_\Theta), \tag{7.6}$$

where

$$L_u(\mathcal{D}'; \mathbf{w}_\mathbf{u}, \mathbf{w}_\Theta) = \mathbb{E}_{\mathcal{D}'}\Big(\|\hat{\mathbf{\Theta}}(\mathbf{x}_{t+1}^{'}; \mathbf{w}_\Theta)(\tilde{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1}^{'}))\| - \|\hat{\mathbf{\Theta}}(\mathbf{x}_t; \mathbf{w}_\Theta)\Delta\mathbf{x}_t\| \Big).$$

Loss function (7.7) ensures that the region of interest $\mathcal{X}$ is contractive with respect to $\hat{\mathbf{\Theta}}$ and the learned dynamics model f'. However, so far there has been no mechanism to ensure that the unique equilibrium of the contractive system is indeed the desired reference value $\mathbf{x}^r$. To alleviate this, we need the learning process to be aware of the desired reference value, which we would like to be the equilibrium of the contraction region. The measure of awareness that we introduce is based on the ability of the controller $\hat{\mathbf{u}}(\mathbf{x}; \mathbf{w}_\mathbf{u})$ to steer the system from an initial state $\mathbf{x}_0 \in \mathcal{X}$ to the desired state value $\mathbf{x}^r$ in k time steps, i.e. how close $\mathbf{x}_k^{'}$ gets to $\mathbf{x}^r$. Therefore, to enforce the system's states to contract to $\mathbf{x}^r$, we add another penalty term to our loss function to obtain the final loss function utilized for learning the policy and contraction metric:

$$L(\mathcal{D}', \mathcal{Y}; \mathbf{w}_\Theta, \mathbf{w}_\mathbf{u}) = L_u(\mathcal{D}'; \mathbf{w}_\Theta, \mathbf{w}_\mathbf{u}) + \alpha L_{\text{tr}}(\mathcal{Y}; \mathbf{w}_\mathbf{u}), \tag{7.7}$$

where

$$L_{\text{tr}}(\mathcal{Y}; \mathbf{w}_\mathbf{u}) = \sum_{\mathbf{x}_0 \in \mathcal{Y}} \left\| \mathbf{x}_k^{'}(\mathbf{x}_0) - \mathbf{x}^r \right\|, \tag{7.8}$$

is the tracking loss with $\alpha \in \mathbb{R}_{>0}$ as the penalty factor. Here, $\mathbf{x}_k^{'}(\mathbf{x}_0)$ is the k^{th} state value

Algorithm 9 Learning Deep Contraction Policies

Input:
  Data set: $(\mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{u}_t) \in \mathcal{D}$
  Set of sampled initial states : $\mathbf{x}_0 \in \mathcal{Y}$
  Reference state: $\mathbf{x}^r$
Init:
  $f^{'}(\mathbf{x}_t, \mathbf{u}_t) \leftarrow$ learned dynamics using $\mathcal{D}$
  $\mathbf{w}_\Theta, \mathbf{w}_\mathbf{u} \leftarrow$ randomly sampled
for N_{epochs} **do**
  Calculate $\mathbf{x}_k^{'}$'s using $\mathcal{Y}$ and $f^{'}(\mathbf{x}_t, \mathbf{u}(\mathbf{x}_t; \mathbf{w}_f))$
  Calculate $L_{\text{tr}}(\mathbf{w}_\mathbf{u})$ using $\mathbf{x}_0 \in \mathcal{Y}$ and $\mathbf{x}_k$'s
  $\Delta\mathbf{x}_t \leftarrow$ uniform random sample from Δ_t
  Create $\mathcal{D}'$ using sampled $\Delta\mathbf{x}_t$
  Calculate $L_u(\mathbf{w}_\Theta, \mathbf{w}_\mathbf{u})$ using $\Delta\mathbf{x}_t$'s and $\mathcal{D}'$
  $L(\mathbf{w}_\Theta, \mathbf{w}_\mathbf{u}) \leftarrow L_u(\mathbf{w}_\Theta, \mathbf{w}_\mathbf{u}) + \alpha L_{\text{tr}}(\mathbf{w}_\mathbf{u})$
  Calculate gradients $\nabla_{\mathbf{w}_\Theta} L$ and $\nabla_{\mathbf{w}_\mathbf{u}} L$
  Update $\mathbf{w}_\Theta$ and $\mathbf{w}_\mathbf{u}$
end for

of the process $\mathbf{x}_{t+1}^{'} = f^{'}(\mathbf{x}_t^{'}, \hat{\mathbf{u}}(\mathbf{x}_t^{'}; \mathbf{w}_\mathbf{u}))$, initialized at $\mathbf{x}_0^{'} = \mathbf{x}_0$ where $\mathbf{x}_0$ is drawn from a countable set $\mathcal{Y} \in \mathcal{X}$. The number of time steps k is set by the designer and, as the reader may infer, affects the transient behavior of the closed-loop system.

7.3 Contraction of True Dynamics Under Learned Policy

A major concern regarding control policy design using a learned model from offline data is that of model mismatch. In order to bound the controller performance degradation, we assume a known upper bound on the Lipschitz constant of the model error $f(\mathbf{x}_t, \mathbf{u}_t) - f^{'}(\mathbf{x}_t, \mathbf{u}_t)$, which we denote as $\mathsf{L}_{f-f'}$. In practice, such an upper bound may be estimated by fitting a Reverse Weibull distribution over the data set $\mathcal{D}$ [145, 146].

Lemma 7.3.1. *Consider an unknown system $f(\mathbf{x}, \mathbf{u})$ and its learned model $f^{'}(\mathbf{x}, \mathbf{u})$ with an upper-bound estimation on the Lipschitz constant of $f(\mathbf{x}, \mathbf{u}) - f^{'}(\mathbf{x}, \mathbf{u})$ as $\mathsf{L}_{f-f'}$. The error*

between the learned model and the unknown system is bounded by ε*, i.e.* $\|f(\mathbf{x},\mathbf{u}) - f^{'}(\mathbf{x},\mathbf{u})\| < \varepsilon$*, for all* $(\mathbf{x},\mathbf{u}) \in \mathcal{X} \times \mathcal{U}$ *where*

$$\varepsilon = \max_{(\mathbf{x}_t,\mathbf{x}_{t+1},\mathbf{u}_t)\in\mathcal{D}} \|\mathbf{x}_{t+1} - f^{'}(\mathbf{x}_t,\mathbf{u}_t)\| + \mathsf{L}_{f-f'}\mathsf{D} \tag{7.9}$$

with $\mathsf{D} = \max_{(\mathbf{x},\mathbf{u})\in\mathcal{X}\times\mathcal{U}} \min_{(\mathbf{x}_t,\mathbf{u}_t)\in\mathcal{D}} \left\| \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} - \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} \right\|.$

Proof. We ground our error analysis on the training error of the tuples $(\mathbf{x}_t,\mathbf{u}_t) \in \mathcal{D}$ and propagate the error to the general state and control tuples $(\mathbf{x},\mathbf{u}) \in \mathcal{X} \times \mathcal{U}$.

$$\begin{aligned}
&\|f(\mathbf{x},\mathbf{u}) - f^{'}(\mathbf{x},\mathbf{u})\| \leq \|f(\mathbf{x}_t,\mathbf{u}_t) - f^{'}(\mathbf{x}_t,\mathbf{u}_t)\| + \\
&\|(f(\mathbf{x},\mathbf{u}) - f^{'}(\mathbf{x},\mathbf{u})) - (f(\mathbf{x}_t,\mathbf{u}_t) - f^{'}(\mathbf{x}_t,\mathbf{u}_t))\| \leq \\
&\|f(\mathbf{x}_t,\mathbf{u}_t) - f^{'}(\mathbf{x}_t,\mathbf{u}_t)\| + \mathsf{L}_{f-f'} \left\| \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} - \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} \right\|.
\end{aligned}$$

The first and the second inequalities are obtained by adding and subtracting the terms $f(\mathbf{x}_t,\mathbf{u}_t)$ and $f^{'}(\mathbf{x}_t,\mathbf{u}_t)$, and also using the norm and Lipschitz constant properties. If we define $E(\mathbf{x}_t,\mathbf{u}_t,\mathbf{x},\mathbf{u})$ as the right hand side of the second inequality, then

$$\max_{(\mathbf{x},\mathbf{u})\in\mathcal{X}\times\mathcal{U}} \min_{(\mathbf{x}_t,\mathbf{u}_t)\in\mathcal{D}} E(\mathbf{x}_t,\mathbf{u}_t,\mathbf{x},\mathbf{u}) \leq \varepsilon$$

where

$$\varepsilon = \max_{(\mathbf{x}_t,\mathbf{x}_{t+1},\mathbf{u}_t)\in\mathcal{D}} \left\|\mathbf{x}_{t+1} - f^{'}(\mathbf{x}_t,\mathbf{u}_t)\right\| + \mathsf{L}_{f-f'}\mathsf{D},$$

which concludes the proof. □

The constant D in Lemma 7.3.1 is the maximum distance that a point $(\mathbf{x},\mathbf{u}) \in \mathcal{X} \times \mathcal{U}$ can have from its nearest data point $(\mathbf{x}_t,\mathbf{u}_t) \in \mathcal{D}$.

Deep contraction policy learning proposed in Algorithm 9 ideally ensures contractive behavior of the controlled learned system $f^{'}(\mathbf{x}_t, \mathbf{u}(\mathbf{x}_t))$ at the states $\mathbf{x}_t \in \mathcal{D}$. More specifically, by defining an approximate measure of contraction condition (1.35) as $C_{g(\mathbf{x}_t)} : \mathcal{X} \times \Delta_t \to \mathbb{R}$

$$C_{g(\mathbf{x}_t)}(\mathbf{x}_t,\Delta\mathbf{x}_t) = \|\hat{\boldsymbol{\Theta}}(g(\mathbf{x}_t))(g(\tilde{\mathbf{x}}_t) - g(\mathbf{x}))\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t)\Delta\mathbf{x}_t\|,$$

the controlled learned model being contractive is equivalent to $\mathbb{E}_{\Delta\mathbf{x}_t}\big(C_{f^{'}(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)\big) < 0$ for all $\mathbf{x}_t \in \mathcal{D}$ and $\Delta\mathbf{x}_t \in \Delta_t$. Hence, it remains for us to verify whether the learned policy exhibits contraction with the true unknown system dynamics in the sense of contraction condition (1.35), i.e. $\mathbb{E}_{\Delta\mathbf{x}_t}\big(C_{f(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)\big) < 0$ for all $\mathbf{x}_t \in \mathcal{X}$ and $\Delta\mathbf{x}_t \in \Delta_t$. We seek to derive a condition under which we are guaranteed that the controlled true dynamics are also contractive with the learned policy. To arrive to such quantification, we begin with contraction of the learned model $C_{f^{'}(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)$ at the training points, $\mathbf{x}_t \in \mathcal{D}$ and end with an upper bound estimation of the contraction of the true dynamics $C_{f(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)$ at any points $\mathbf{x} \in \mathcal{X}$. The following Proposition establishes the condition under which the approximate contraction measure holds for the true robot dynamics under the trained $\mathbf{u}(\mathbf{x}_t)$.

Proposition 7.3.1. *Let* $\mathbb{E}_{\Delta\mathbf{x}_t}\big(C_{f^{'}(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)\big) < 0$ *for all* $\mathbf{x}_t \in \mathcal{D}$. *Let the Lipschitz constant of* $\hat{\boldsymbol{\Theta}}_{ij}(\mathbf{x}_t)$, $f(\mathbf{x}_t, \mathbf{u}_t) - f^{'}(\mathbf{x}_t, \mathbf{u}_t)$, $f^{'}(\mathbf{x}_t, \mathbf{u}_t)$, $\hat{\mathbf{u}}(\mathbf{x}_t)$, $f^{'}(\mathbf{x}_t, \hat{\mathbf{u}}(\mathbf{x}_t))$ *and* $C_{f^{'}(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)$ *be given as* $\mathsf{L}_{\boldsymbol{\Theta}_{ij}}$, $\mathsf{L}_{f-f'}$, $\mathsf{L}_{f'}$, $\mathsf{L}_{\mathbf{u}}$, $\mathsf{L}_{f'_u}$, *and* L_C, *respectively. Additionally, let* $|\hat{\boldsymbol{\Theta}}_{ij}(\mathbf{x}_t)| < \gamma$, λ *be given by* (7.3), *and* ε *be given as in* (7.9). *Then the true dynamics* (7.1) *are contractive under the trained controlled policy* $\hat{\mathbf{u}}(\mathbf{x}_t)$, *i.e.* $\mathbb{E}_{\Delta\mathbf{x}_t}\big(C_{f(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t)\big) < 0$ *for all* $\mathbf{x}_t \in \mathcal{X}$ *and* $\Delta\mathbf{x}_t \in \Delta_t$, *if*

$$\zeta + \lambda(\varepsilon\tau\mathsf{L}_{f'_u} + (\varepsilon\tau + n\gamma)\mathsf{L}_{f-f'}(1 + \mathsf{L}_{\mathbf{u}})) < 0, \tag{7.10}$$

where $\tau = \sqrt{\sum_{ij} \mathsf{L}^2_{\boldsymbol{\Theta}_{ij}}}$ *and* $\zeta = \max_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{x}_t\in\mathcal{D}} C(\mathbf{x}_t) + \mathsf{L}_C \|\mathbf{x}_t - \mathbf{x}\|$.

Proof. We want to derive a sufficient condition which ensures that contraction condition (7.4)

holds for the true dynamics model. Using the learned dynamics model, the left-hand side of (7.4) for $\mathbf{x}_t \in \mathcal{X}$ can be bounded for the true dynamics as

$$
\begin{aligned}
&\|\hat{\boldsymbol{\Theta}}(\mathbf{x}_{t+1})(\tilde{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t)\Delta\mathbf{x}_t\| \leq \\
&\|\hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1})(\tilde{\mathbf{x}}^{'}_{t+1} - \mathbf{x}^{'}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t)\Delta\mathbf{x}_t\| + \\
&\|(\hat{\boldsymbol{\Theta}}(\mathbf{x}_{t+1}) - \hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1}))(\tilde{\mathbf{x}}^{'}_{t+1} - \mathbf{x}^{'}_{t+1})\| + \\
&\|\hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1})((\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}^{'}_{t+1}) - (\mathbf{x}_{t+1} - \mathbf{x}^{'}_{t+1}))\| + \\
&\|(\hat{\boldsymbol{\Theta}}(\mathbf{x}_{t+1}) - \hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1}))((\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}^{'}_{t+1}) - (\mathbf{x}_{t+1} - \mathbf{x}^{'}_{t+1}))\|,
\end{aligned}
$$

where $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \hat{\mathbf{u}}(\mathbf{x}_t))$, $\tilde{\mathbf{x}}_{t+1} = f(\tilde{\mathbf{x}}_t, \hat{\mathbf{u}}(\tilde{\mathbf{x}}_t))$, $\mathbf{x}^{'}_{t+1} = f^{'}(\mathbf{x}_t, \hat{\mathbf{u}}(\mathbf{x}_t))$, and $\tilde{\mathbf{x}}^{'}_{t+1} = f^{'}(\tilde{\mathbf{x}}_t, \hat{\mathbf{u}}(\tilde{\mathbf{x}}_t))$. The inequality holds due to addition and subtraction of proper terms and norm properties. The inequality can be further simplified using the Frobenius norm of the contraction metric $\hat{\boldsymbol{\Theta}}$. Since, by assumption, the entries of the contraction metric are bounded by γ, we have $\|\hat{\boldsymbol{\Theta}}(\mathbf{x})\|_F \leq n\gamma$. Having an upper bound estimate of the Lipschitz constant of entries of the contraction metric $\mathsf{L}_{\boldsymbol{\Theta}_{ij}}$ and recalling that $\|(\tilde{\mathbf{x}}^{'}_{t+1} - \mathbf{x}^{'}_{t+1})\| \leq \varepsilon$ from Lemma 7.3.1, leads to the result $\|(\hat{\boldsymbol{\Theta}}(\mathbf{x}_{t+1}) - \hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1}))\|_F \leq \varepsilon\sqrt{\sum_{ij}\mathsf{L}^2_{\boldsymbol{\Theta}_{ij}}}$. In addition, using the estimated Lipschitz constant $\mathsf{L}_{f-f'}$, we have that $\|((\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}^{'}_{t+1}) - (\mathbf{x}_{t+1} - \mathbf{x}^{'}_{t+1}))\| \leq \mathsf{L}_{f-f'}\left\|\begin{bmatrix}\tilde{\mathbf{x}} \\ \mathbf{u}(\tilde{\mathbf{x}})\end{bmatrix} - \begin{bmatrix}\mathbf{x} \\ \mathbf{u}(\mathbf{x})\end{bmatrix}\right\|$. Now, using the Lipschitz constant of $\mathbf{u}(\mathbf{x})$ as $\mathsf{L}_{\mathbf{u}}$, we have that $\|((\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}^{'}_{t+1}) - (\mathbf{x}_{t+1} - \mathbf{x}^{'}_{t+1}))\| \leq \mathsf{L}_{f-f'}\lambda(1 + \mathsf{L}_{\mathbf{u}})$. Finally, we can write the following inequality:

$$
\begin{aligned}
&\|\hat{\boldsymbol{\Theta}}(\mathbf{x}_{t+1})(\tilde{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t)\Delta\mathbf{x}_t\| \leq \\
&\|\hat{\boldsymbol{\Theta}}(\mathbf{x}^{'}_{t+1})(\tilde{\mathbf{x}}^{'}_{t+1} - \mathbf{x}^{'}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t)\Delta\mathbf{x}_t\| + \\
&\lambda(\varepsilon\tau\mathsf{L}_{f^{'}_u} + (\varepsilon\tau + n\gamma)\mathsf{L}_{f-f'}(1 + \mathsf{L}_{\mathbf{u}})),
\end{aligned} \tag{7.11}
$$

where $\tau = \sqrt{\sum_{ij}\mathsf{L}^2_{\boldsymbol{\Theta}_{ij}}}$. With Lipschitz constant L_C, we can derive an upper bound for

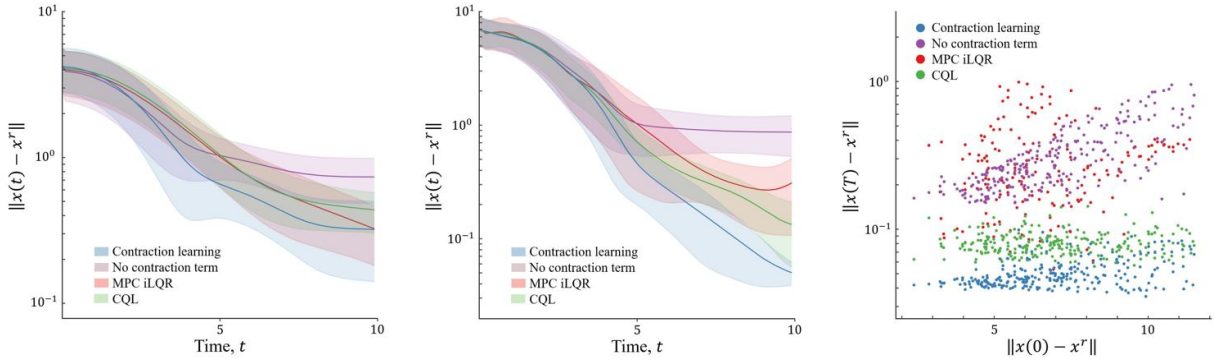


Figure 7.2 – Norm of the tracking error over a collection of 256 initial states for the 2D car problem (left) and the 3D drone problem (middle). The y axis is shown on a logarithmic scale and results capture the mean plus and minus one standard deviation. We see that the additional complexity of the 3D drone over the 2D car model allows for greater variation in algorithm performance. **N**orm of the average final tracking error versus the norm of the initial tracking error (right). As the initial states approach the boundary of the region of interest, controller performance tends to degrade.

$C_{f'(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}, \Delta\mathbf{x}_t)$, $\mathbf{x}_t \in \mathcal{X}$ and $\Delta\mathbf{x}_t \in \Delta_t$, such that $C_{f'(\mathbf{x},\hat{\mathbf{u}}(\mathbf{x}))}(\mathbf{x}_t, \Delta\mathbf{x}_t) < \zeta$ where

$$\zeta = \max_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{x}_t\in\mathcal{D}} C_{f'(\mathbf{x}_t,\hat{\mathbf{u}}(\mathbf{x}_t))}(\mathbf{x}_t, \Delta\mathbf{x}_t) + \mathsf{L}_C\|\mathbf{x}_t - \mathbf{x}\|$$

. Finally by taking the expectation on Equation (7.11), we get

$$\mathbb{E}_{\Delta\mathbf{x}_t}\left(\|\hat{\boldsymbol{\Theta}}(\mathbf{x}_{t+1})(\tilde{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1})\| - \|\hat{\boldsymbol{\Theta}}(\mathbf{x}_t)\Delta\mathbf{x}_t\|\right) \leq$$
$$\zeta + \lambda(\varepsilon\tau\mathsf{L}_{f'_u} + (\varepsilon\tau + n\gamma)\mathsf{L}_{f-f'}(1 + \mathsf{L}_{\mathbf{u}}))$$

which concludes the proof. □

7.4 Implementation and Evaluation

We evaluate the performance of the contraction policies in a set of goal-reaching robotic tasks by comparing our method against a number of offline control methods suitable for systems with learned dynamics models. In particular, we compare our framework with the following

algorithms:

1. **MPC:** An iterative Linear Quadratic Controller (iLQR) as described in [147] ran in a receding horizon fashion.
2. **Learning without contraction:** To evaluate the effectiveness of the contraction penalty, we further evaluate the robot's performance in the absence of any contraction terms in the loss function.
3. **Reinforcement Learning:** We also use the state-of-the-art offline RL method Conservative Q-Learning (CQL) [148] for further comparisons.

We evaluate the performance of our approach on two different robotic settings involving nonlinear dynamical systems of varying complexity represented by neural networks. The dynamics of these systems have closed-form expressions, but it is assumed that we do not have access to such expressions. We assume that we only have access to a set of system trajectories and learn a dynamics model from the state-action trajectories. The learned dynamics are represented as neural networks to the model-based control methods: deep contraction policy, MPC controller, and contraction-free learning. The RL implementation develops the policy directly from the same offline data set that is used to train the dynamics model in a model-free fashion. This allows us to implement our algorithm on the learned systems while having an analytical baseline to compare against to quantify robustness. Additionally, we consider state and control sets $\mathcal{X}$,$\mathcal{U}$ defined by box constraints in order to constrain the size of the training data. Clearly for such constraints, $\mathcal{X}$ is convex. The dynamical systems we have chosen for our performance evaluation are as follows:

1. **2D Planar Car:** A planar vehicle that is capable of controlling its acceleration, α, and angular velocity, ω. Here $\mathbf{x} := [p_x, p_y, \theta, v]$ and $\mathbf{u} := [\alpha, \omega]$ where p_x, p_y are the

planar positions, v is the velocity, and θ is the heading angle. The system dynamics are governed by: $\dot{\mathbf{x}} = [v\cos(\theta), v\sin(\theta), \omega, \alpha]^\top$.

2. **3D Drone:** An adaptation of a drone model that is given by [100] and [149]. This model describes an aerial vehicle capable of directly controlling the rate of change of it's normalized thrust $\dot{F}$, and Euler Angles, $\dot{\phi}, \dot{\theta}, \dot{\psi}$. Here $\mathbf{x} := [p_x, p_y, p_z, v_x, v_y, v_z, F, \phi, \theta, \psi]$ and $\mathbf{u} := [\dot{\phi}, \dot{\theta}, \dot{\psi}]$ where p_i, v_i are the translational positions and velocities along the i_{th} axis, respectively. Omitting the first order integrators in $p_i, F, \phi, \theta, \psi$ for brevity, the dynamics can then be expressed as $[\dot{v}_x, \dot{v}_y, \dot{v}_z] = [-F\sin(\theta), F\cos(\theta)\sin(\phi), g - \cos(\theta)\cos(\phi)]$, where g is the acceleration due to gravity.

For both systems we assume a timestep of $\Delta t = 0.1$s and a final time of $T = 10$s.

7.4.1 Learning System Dynamics

All of the continuous dynamical systems described above are represented to our controllers as fully connected neural networks which capture the discretization of the model integration: $\mathbf{x}_{t+1} - \mathbf{x}_t \approx f'(\mathbf{x}_t, \mathbf{u}_t; \mathbf{w}_f)$. The training dataset $\mathcal{D}$ is generated by aggregating reference trajectories through the state space generated from an iLQR controller applied directly to the true dynamics $f(\mathbf{x}_t, \mathbf{u}_t)$. The reference trajectories $\Phi(\mathbf{x}_t, \mathbf{u}_t)$ were chosen such that $\mathbf{x}_t \in \mathcal{X}$ and $\mathbf{u}_t \in \mathcal{U}$ for all t. Trajectory data was used in order to implement a discounted multistep prediction error as in [150] until sufficient integration accuracy was achieved.

7.4.2 Controller Implementation

The contraction metric and control policy neural networks, $\hat{\mathbf{\Theta}}(\mathbf{x}_t; \mathbf{w}_\Theta)$ and $\hat{\mathbf{u}}(\mathbf{x}_t; \mathbf{w}_\mathbf{u})$, are trained according to Algorithm 9.

For our ablation study, we remove the contraction penalty term and simply find a policy for minimizing the tracking error norm. Without a contraction penalty, the impact of contraction conditions during the learning process vanishes. In order to create a controller for this case, each $\mathbf{x}_t \in \mathcal{D}$ is forward evolved a number of time steps under the learned control policy and trained with a discounted cumulative loss of the tracking error norms over each timestep.

For the MPC controller, the iLQR planner utilizes the learned dynamics model in order to calculate the linearization relative to the state and control inputs. This linearization is used along with weighting matrices $\mathbf{Q} = 100\mathbf{I}, \mathbf{R} = 1000\mathbf{I}$ in order to calculate an iLQR control law.

In order to train an offline reinforcement learning algorithm like CQL, the algorithm needs access to state, action, and reward pairs. We reutilize the offline iLQR trajectories created for dynamics learning as training episodes for the offline CQL RL algorithm. The reward at each time step is taken to be the negative norm of the tracking error at the next time step given the currently taken action.

Table 7.1 – Tracking Error Norm RMSE, 3D Drone

Dynamics Model	Test Loss	Contraction learning	No contraction term	MPC iLQR
1	5.67e-05	$\mathbf{1.905 \pm 0.651}$	2.391 ± 0.968	2.161 ± 0.913
2	8.19e-05	$\mathbf{2.026 \pm 0.676}$	2.528 ± 0.880	2.966 ± 1.593
3	1.14e-04	$\mathbf{2.214 \pm 0.889}$	3.315 ± 0.917	6.458 ± 2.284
4	1.58e-04	$\mathbf{2.891 \pm 1.061}$	5.392 ± 1.290	N/A*
5	2.64e-04	$\mathbf{3.571 \pm 1.252}$	N/A*	N/A*

*Values of N/A represent cases where sufficiently stabilizing controllers were not generated.

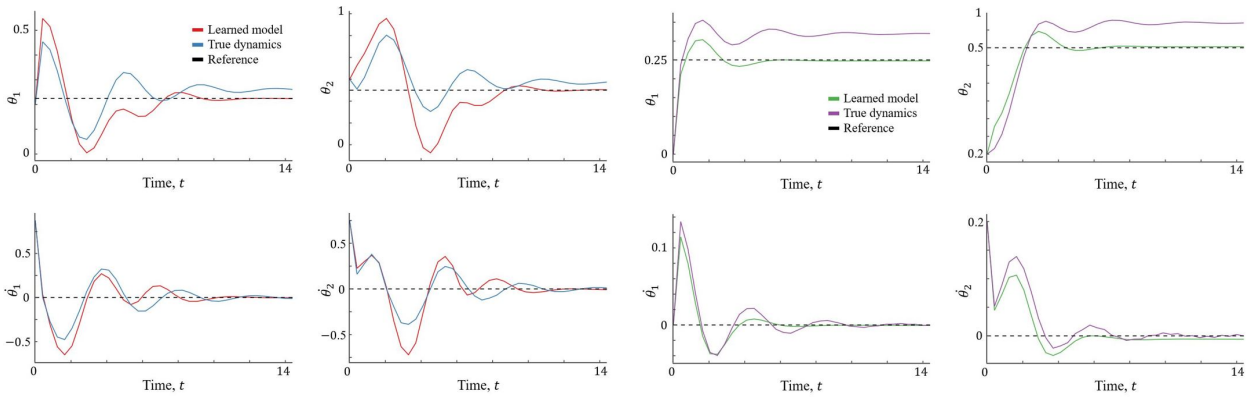


Figure 7.3 – Angular position and angular velocity of the double pendulum system. The controlled learned model and controlled true dynamics are shown. Results are shown for two different sets of initial conditions

7.4.3 Performance Results

In order to compare the performance of our method with the alternative implementations outlined above, we propose a number of metrics to compare the controllers:

- The time evolution of the tracking error, to quantify the controllers' ability to converge to the desired reference $\mathbf{x}^r$.
- The converged tracking error versus the initial tracking error, to quantify the controllers' ability to operate over the working space $\mathcal{X} \times \mathcal{U}$.
- Root Mean Square Error (RMSE) of the tracking error versus learned model loss, to quantify the controllers' ability to deal with model mismatch.

For all analyses, the controllers were each presented with an identical set of 256 initial conditions within $\mathcal{X}$. The control methods were implemented as described above in an attempt to drive these initial states to the desired reference $\mathbf{x}^r$. The results were aggregated over the 256 initial conditions for the 2D car and 3D drone.

In the time evolution analysis, desirable controllers have trajectories that quickly converge, have minimal tracking error norm, and have high convergence precision. Results directly

comparing all of the controllers relative to this performance measure for the two dynamical systems are given in Figure 7.2 (left and middle). The results show that over the two different systems and a multitude of initial conditions, the contraction learning policy performs well relative to the proposed comparison controllers. For the simpler dynamic system of the two, the 2D planar car, the results are comparable among all controllers but favor the contraction controller, while the more complex drone environment shows the clear benefits of our approach. The enforcement of contraction conditions forces nearby trajectories to converge to one another, and when near the reference point, this has the effect of reducing the norm of the tracking error further than the systems designed without contraction in mind. The contraction controller consistently has the lowest mean norm of the tracking error over all the sampled initial states.

Comparing the converged tacking error, in this case, the average of the final 10 timestamps of each trajectory, versus the initial tracking error gives insight into the performance of the controllers' over the entire state and control space $\mathcal{X}$ and $\mathcal{U}$. Cases with a higher initial tracking error represent trajectories that start closer towards the boundary of our working space $\mathcal{X} \times \mathcal{U}$. Favorable controllers are ones in which the converged tracking error remains constant or grows slowly as the initial tracking error increases. Figure 7.2 (right) directly shows this comparison. The results here clearly show that the MPC controller and the learned policy without the contraction terms have difficulty as the initial state norm gets further from the desired reference. For the MPC controller, the poor performance is likely caused by not having expressive enough dynamics due to the repeated linearization process. The contraction-free policy shows good performance for small initial tracking errors but quickly degrades as this value grows. Such a control method acts extremely locally. Training a collection of states to converge to the reference without the additional contraction structure does not yield favorable stability properties. The CQL policy and deep contraction learning generate trajectories with minimal degradation as the tracking error increases, with the contraction learning method consistently having the highest degree of performance.

Finally, for the 3D drone scenario, the impact of the learned dynamics model quality on the model-based controllers' performance is studied. To this end, multiple models of different quality were learned from the same offline data set. Since the CQL policy is directly learned from the offline data and does not utilize the learned dynamics model, this method is omitted from this analysis. In this case, favorable controllers are ones in which the error grows slowly with increasing model inaccuracy. Comparison of the RMSE values of the tracking error norm over the length of the trajectories for the varying quality dynamics models are shown in Table 7.1. The contraction learning model shows favorable performance as dynamics model mismatch increases due to the robustness properties discussed in Section **??**. For particularly low-quality learned dynamics models we even see that the deep contraction policy is able to generate stabilizing controllers where the contraction-free policy and MPC controller fail to do so.

7.4.4 Non-control Affine Analysis

In order to quantify the ability of our deep contraction policy learning to generalize to more complex systems, we perform an illustrative analysis of our controller on the double pendulum model given in [151]. Such a system is chaotic with a non-affine control input. Figure 7.3 shows the comparison of two scenarios where the designed controller was implemented on both the learned model and the true dynamics. The trajectories show the controller is able to accomplish the task when applied to the learned model. However, when applied to the true dynamics, the controller positions the arms with a slight positional error while keeping the angular velocity at zero.

7.5 Conclusion

In this chapter, we established a framework for learning a converging control policy for an unknown system from offline data. We leveraged Contraction theory and proposed a data augmentation method for encoding the contraction conditions directly into the loss function. We jointly learned the control policy and its corresponding contraction metric. We compared our method with several state-of-the-art control algorithms and showed that our method provides faster convergence, a smaller tracking error, lower variance of trajectories. For our future work, we would like to extend the current work to develop the stochastic confidence bound of our control design approach design.

Chapter 8

Conclusion and Future Work

This dissertation work has focused on the development of distributed strategy selection for networked mobile agents (robots, people, unmanned aerial vehicles, and so on) for challenges in which the aim is to maximize the agents' total utility. The goal is to create a decentralized cooperative decision-making algorithm that does not require any infrastructure and can function well in the face of system uncertainty. The work's principal application is cooperative strategy selection solutions for tasks like dynamic area patrolling, persistent monitoring, sensor placement, and area coverage, where avoiding action overlaps and increasing the agents' performance is difficult. This dissertation project aims to create a decentralized strategy selection algorithm that relies on the agents' local processing power and communication network. To address this problem, we tied a utility function to the combined set of decisions made by the team of agents. By showing that the utility function is a monotone increasing and submodular function, we formulate a general utility maximization problem as maximizing a submodular set function subject to partition matroid. We then proceed to propose a suboptimal strategy selection algorithm with known optimality bound. We work in the value oracle model where the only access of the agents to the utility function is through a black box that returns the utility function value and where the agents can communicate over

a connected undirected graph and have access only to their own strategy set. Our proposed algorithm is based on a distributed stochastic gradient ascent scheme built on the multilinear extension of the submodular set function. We use a maximum consensus protocol to minimize the inconsistency of the shared information over the network caused by a delay in the flow of information while solving for the fractional solution of the multilinear extension model. Furthermore, we propose a distributed framework for finding a set solution using the fractional solution. However, our proposed communication protocol informs adversarial elements on the selected strategies by the agents. Our next contribution is to design a distributed algorithm that enables each agent to find a suboptimal policy locally with a guaranteed level of privacy. We base our modified algorithm's privacy preservation characteristic on our proposed stochastic rounding method and tie the level of privacy to a privacy variable. That is, the policy choice of an agent can be determined with at most a certain probability. We also show that there is an interplay between the level of optimality gap and the guaranteed level of privacy. To address the problem of the interplay between optimality gap and level of privacy we investigated the privacy preservation method by adding permissible perturbation signals to the local dynamics and the agents' broadcast out signals, we study the feasibility of protecting the privacy of the reference value of the agents. Admissible additive perturbation signals are ones that do not deviate from the algorithm's ultimate decision point from when no perturbation signal is introduced. Our findings indicate that if an adversarial agent has access to another agent's output as well as all of the input signals sent to that agent, the adversary can figure out what that agent's secret decision is, independent of the perturbation signals. Otherwise, the agent's privacy can be protected. Our proposed method is targeting networks that serve the average consensus algorithms. In future work, the goal will be to redesign our privacy-preserving method to be compatible with maximum consensus decision-making protocols. In some cases, our suggested randomized distributed strategy selection may need an excessive amount of samples. To solve this challenge, we consider reusing offline data and employing neural networks to represent the system. Neural

networks, on the other hand, are known for being unpredictable, making them unsuitable for use in dynamic systems. We developed some preliminary results to address the problem of uncertainty, we suggested a way of certifying neural networks in the context of dynamical systems and policy-making utilizing contraction theory. However, our proposed scheme does not directly address the reuse of samples in randomized distributed strategy selection and multi-linear extension of a submodular set function. In future work, we suggest tailoring our contraction theory-based approach to conform with our proposed randomized distributed strategy selection scheme.

Bibliography

[1] C. Chekuri and A. Kumar, "Maximum coverage problem with group budget constraints and applications," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 72–83, Springer, 2004.

[2] A. Robey, A. Adibi, B. Schlotfeldt, J. G. Pappas, and H. Hassani, "Optimal algorithms for submodular maximization with distributed constraints," *arXiv preprint arXiv:1909.13676*, 2019.

[3] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Tran. on Automatic Control*, vol. 62, no. 2, pp. 753–765, 2017.

[4] J. Vondrák, "Optimal approximation for the submodular welfare problem in the value oracle model," in *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pp. 67–74, 2008.

[5] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Tran. on Automatic Control*, vol. 20, no. 2, pp. 243–255, 2004.

[6] A. Krause and C. Guestrin, "Near-optimal observation selection using submodular functions," in *American Association for Artificial Intelligence*, vol. 7, pp. 1650–1654, 2007.

[7] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, no. Feb, pp. 235–284, 2008.

[8] M. Schwager, D. Rus, and J. Slotine, "Decentralized, adaptive coverage control for networked robots," *The Int. Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.

[9] F. Bullo, R. Carli, and P. Frasca, "Gossip coverage control for robotic networks: Dynamical systems on the space of partitions," *SIAM Journal on Control and Optimization*, vol. 50, no. 1, pp. 419–447, 2012.

[10] A. Carron, M. Todescato, R. Carli, L. Schenato, and G. Pillonetto, "Multi-agents adaptive estimation and coverage control using gaussian regression," in *2015 European Control Conference (ECC)*, pp. 2490–2495, IEEE, 2015.

[11] M. Todescato, A. Carron, R. Carli, G. Pillonetto, and L. Schenato, "Multi-robots gaussian estimation and coverage control: From client–server to peer-to-peer architectures," *Automatica*, vol. 80, pp. 284–294, 2017.

[12] Y. Chung and S. S. Kia, "A distributed service-matching coverage via heterogeneous mobile agents," 2020. available at `https://arxiv.org/pdf/2009.11943.pdf`.

[13] C. Yuan, Y. Zhang, and Z. Liu, "A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques," *Canadian Journal of Forest Research*, vol. 45, no. 7, pp. 783–792, 2015.

[14] N. Henry and O. Henry, "Wireless sensor networks based pipeline vandalisation and oil spillage monitoring and detection: main benefits for nigeria oil and gas sectors," *The SIJ Tran. on Computer Science Engineering & its Applications (CSEA)*, vol. 3, no. 1, pp. 1–6, 2015.

[15] R. Engler, A. Guisan, and L. Rechsteiner, "An improved approach for predicting the distribution of rare and endangered species from occurrence and pseudo-absence data," *J. of Applied Ecology*, vol. 41, no. 2, pp. 263–274, 2004.

[16] T. Thomas and E. van Berkum, "Detection of incidents and events in urban networks," *IET Intelligent Transport Systems*, vol. 3, no. 2, pp. 198–205, 2009.

[17] R. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*, pp. 85–103, Springer, 1972.

[18] A. Machado, G. Ramalho, J. Zucker, and A. Drogoul, "Multi-agent patrolling: An empirical analysis of alternative architectures," in *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pp. 155–170, 2002.

[19] A. Almeida, G. Ramalho, H. Santana, P. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyre, "Recent advances on multi-agent patrolling," in *Brazilian Symposium on Artificial Intelligence*, pp. 474–483, Springer, 2004.

[20] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *Intelligent Agent Technology*, pp. 302–308, IEEE, 2004.

[21] F. Pasqualetti, A. Franchi, and F. Bullo, "On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms," *IEEE Tran. on Robotics*, vol. 28, no. 3, pp. 592–606, 2012.

[22] J. Yu, S. Karaman, and D. Rus, "Persistent monitoring of events with stochastic arrivals at multiple stations," *IEEE Tran. on Robotics*, vol. 31, no. 3, pp. 521–535, 2015.

[23] M. Donahue, G. Rosman, K. Kotowick, D. Rus, and C. Baykal, "Persistent surveillance of transient events with unknown statistics," tech. rep., MIT Lincoln Laboratory Lexington United States, 2016.

[24] A. Asghar, S. Smith, and S. Sundaram, "Multi-robot routing for persistent monitoring with latency constraints," *arXiv preprint arXiv:1903.06105*, 2019.

[25] A. Farinelli, L. Iocchi, and D. Nardi, "Distributed on-line dynamic task assignment for multi-robot patrolling," *Autonomous Robots*, vol. 41, no. 6, pp. 1321–1345, 2017.

[26] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan, "The minimum latency problem," in *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, pp. 163–171, 1994.

[27] Z. Kadelburg, D. Dukic, M. Lukic, and I. Matic, "Inequalities of Karamata, Schur and Muirhead, and some applications," *The Teaching of Mathematics*, vol. 8, no. 1, pp. 31–45, 2005.

[28] T. Summers, F. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks.," *IEEE Tran. on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2016.

[29] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "Submodular optimization for voltage control," *IEEE Tran. on Power Systems*, vol. 33, no. 1, pp. 502–513, 2018.

[30] A. Clark, P. Lee, B. Alomair, L. Bushnell, and R. Poovendran, "Combinatorial algorithms for control of biological regulatory networks," *IEEE Tran. on Control of Network Systems*, vol. 5, no. 2, pp. 748–759, 2018.

[31] A. Clark, L. Bushnell, and R. Poovendran, "A supermodular optimization framework for leader selection under link noise in linear multi-agent systems," *IEEE Tran. on Automatic Control*, vol. 59, no. 2, pp. 283–296, 2014.

[32] L. Fisher, G. Nemhauser, and L. Wolsey, "An analysis of approximations for maximizing submodular set functions—ii," in *Polyhedral combinatorics*, pp. 73–87, Springer, 1978.

[33] N. Mehr and R. Horowitz, "A submodular approach for optimal sensor placement in traffic networks," in *2018 Annual American Control Conference (ACC)*, pp. 6353–6358, IEEE, 2018.

[34] J. Qin, I. Yang, and R. Rajagopal, "Submodularity of storage placement optimization in power networks," *IEEE Tran. on Automatic Control*, vol. 64, no. 8, pp. 3268–3283, 2019.

[35] M. Bucciarelli, S. Paoletti, E. Dall'Anese, and A. Vicino, "On the greedy placement of energy storage systems in distribution grids," in *American Control Conference*, 2020.

[36] S. T. Jawaid and S. Smith, "Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems," *Automatica*, vol. 61, pp. 282–288, 2015.

[37] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "Towards scalable voltage control in smart grid: A submodular optimization approach," in *ACM/IEEE 7th International Conference on Cyber-Physical Systems*, 2016.

[38] N. Rezazadeh and S. S. Kia, "A sub-modular receding horizon solution for mobile multi-agent persistent monitoring," *Automatica*, vol. 127, p. 109460, 2021.

[39] A. Krause and D. Golovin, "Submodular function maximization," in *Tractability: Practical Approaches to Hard Problems* (L. Bordeaux, Y. Hamadi, and P. Kohli, eds.), pp. 71–104, Cambridge, UK: Cambridge University Press, 2014.

[40] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time," *Journal of Combinatorial Theory, Series B*, vol. 8, p. 346–355, 2000.

[41] L. F. S. Fujishige and S. Iwata, "A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions," *Journal of the ACM*, vol. 48, no. 4, pp. 761–777, 2001.

[42] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—i," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[43] L. M. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions—ii," in *Polyhedral Combinatorics*, pp. 73–87, Springer, 1978.

[44] G. L. Nemhauser and L. A. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Math. Operations Research*, vol. 3, no. 3, pp. 177–188, 1978.

[45] M. Conforti and G. Cornuéjols, "Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem," *Discrete applied mathematics*, vol. 7, no. 3, pp. 251–274, 1984.

[46] J. Vondrák, "Submodularity and curvature: The optimal algorithm (combinatorial optimization and discrete algorithms)," 2010.

[47] A. A. Bian, B. Mirzasoleiman, J. Buhmann, and A. Krause, "Guaranteed non-convex optimization: Submodular maximization over continuous domains," in *Artificial Intelligence and Statistics*, pp. 111–120, 2017.

[48] A. Mokhtari, H. Hassani, and A. Karbasi, "Stochastic conditional gradient methods: From convex minimization to submodular maximization," *Journal of Machine Learning Research*, vol. 21, no. 105, pp. 1–49, 2020.

[49] O. Sadeghi and M. Fazel, "Online continuous dr-submodular maximization with long-term budget constraints," in *International Conference on Artificial Intelligence and Statistics*, pp. 4410–4419, 2020.

[50] U. Feige, "A threshold of ln n for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[51] R. Konda, D. Grimsman, and J. Marden, "Execution order matters in greedy algorithms with limited information," *arXiv preprint arXiv:2111.09154*, 2021.

[52] K. Wei, R. Iyer, and J. Bilmes, "Fast multi-stage submodular maximization," in *International conference on machine learning*, pp. 1494–1502, PMLR, 2014.

[53] B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause, "Distributed submodular maximization: Identifying representative elements in massive data," in *Advances in Neural Information Processing Systems*, pp. 2049–2057, 2013.

[54] B. Mirzasoleiman, M. Zadimoghaddam, and A. Karbasi, "Fast distributed submodular cover: Public-private data summarization," in *Advances in Neural Information Processing Systems*, pp. 3594–3602, 2016.

[55] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani, "Fast greedy algorithms in mapreduce and streaming," *ACM Transactions on Parallel Computing*, vol. 2, no. 3, pp. 1–22, 2015.

[56] P. S. Raut, O. Sadeghi, and M. Fazel, "Online dr-submodular maximization with stochastic cumulative constraints," *arXiv preprint arXiv:2005.14708*, 2020.

[57] A. Mokhtari, H. Hassani, and A. Karbasi, "Decentralized submodular maximization: Bridging discrete and continuous settings," in *International Conference on Machine Learning*, pp. 3616–3625, 2018.

[58] J. Xie, C. Zhang, Z. Shen, C. Mi, and H. Qian, "Decentralized gradient tracking for continuous dr-submodular maximization," in *International Conference on Artificial Intelligence and Statistics*, pp. 2897–2906, 2019.

[59] L. Ye and S. Sundaram, "Distributed maximization of submodular and approximately submodular functions," in *IEEE Conference on Decision and Control (CDC)*, pp. 2979–2984, 2020.

[60] A. Ageev and M. Sviridenko, "Pipage rounding: A new method of constructing algorithms with proven performance guarantee," *Journal of Combinatorial Optimization*, vol. 8, no. 3, pp. 307–328, 2004.

[61] N. Rezazadeh and S. Kia, "Multi-agent maximization of a monotone submodular function via maximum consensus," in *IEEE Conference on Decision and Control*, pp. 1238–1243, IEEE, 2021.

[62] N. Rezazadeh and S. S.S. Kia, "Distributed strategy selection: A submodular set function maximization approach," *arXiv preprint arXiv:2107.14371*, 2021.

[63] N. Rezazadeh and S. Kia, "A sub-modular receding horizon solution for mobile multi-agent persistent monitoring," *Automatica*, vol. 127, p. 109460, 2021.

[64] B. Gharesifard and S. Smith, "Distributed submodular maximization with limited information," *IEEE transactions on control of network systems*, vol. 5, no. 4, pp. 1635–1645, 2017.

[65] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization," in *Proceedings of Int. Conference on Distributed Computing and Networking*, p. 4, 2015.

[66] Y. Wang, "Privacy-preserving average consensus via state decomposition," *IEEE Transactions on Automatic Control*, vol. 64, no. 11, pp. 4711–4716, 2019.

[67] M. Mitrovic, M. Bun, A. Krause, and A. Karbasi, "Differentially private submodular maximization: Data summarization in disguise," in *Int. Conference on Machine Learning*, pp. 2478–2487, 2017.

[68] S. Perez-Salazar and R. Cummings, "Differentially private online submodular maximization," in *Int. Conference on Artificial Intelligence and Statistics*, pp. 1279–1287, 2021.

[69] A. Rafiey and Y. Yoshida, "Fast and private submodular and k-submodular functions maximization with matroid constraints," in *Int. Conference on Machine Learning*, pp. 7887–7897, 2020.

[70] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Tran. on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[71] W. Reb and R. W. Beard, "Consensus seeking in multi-agent systems under dynamically changing interaction topologies," *IEEE Tran. on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[72] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems and Control Letters*, vol. 53, pp. 65–78, 2004.

[73] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[74] M. Kefayati, M. S. Talebi, B. H. Khalaj, and H. R. Rabiee, "Secure consensus averaging in sensor networks using random offsets," in *International Conference on Telecommunications*, pp. 556–560, 2007.

[75] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: obstructions, trade-offs, and optimal algorithm design," *Automatica*, vol. 81, pp. 221–231, 2017.

[76] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *IEEE Symposium on Foundations of Computer Science, 48th Annual*, pp. 94–103, 2007.

[77] A. Friedman and A. Schuster, "Data mining with differential privacy," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 493–502, 2010.

[78] C. Dwork, "Differential privacy: A survey of results," in *International Conference on Theory and Applications of Models of Computation*, pp. 1–19, 2008.

[79] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[80] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *European Control Conference*, pp. 760–765, 2013.

[81] J. Le Ny and G. Pappas, "Differentially private Kalman filtering," in *Allerton Conf. on Communications, Control and Computing*, pp. 1618–1625, 2012.

[82] J. Le Ny and G. J. Pappas, "Differential private filtering," *IEEE Tran. on Automatic Control*, vol. 59, no. 2, pp. 341–354, 2014.

[83] J. Cortes, G. E. Dullerud, S. Han, J. L. Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," in *IEEE Int. Conf. on Decision and Control*, pp. 4252–4272, 2016.

[84] M. Ruan, M. Ahmad, and Y. Wang, "Secure and privacy-preserving average consensus," in *ACM Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and Privacy*, pp. 123–129, 2017.

[85] A. Esteki and S. S. Kia, "Deterministic privacy preservation in static average consensus problem," *IEEE Control Systems Letters*, vol. 5, no. 6, pp. 2036–2041, 2020.

[86] S. S. Kia, J. Cortés, and S. Martínez, "Dynamic average consensus under limited control authority and privacy requirements," *International Journal on Robust and Nonlinear Control*, vol. 25, no. 13, pp. 1941–1966, 2015.

[87] E. D. Sontag, "Input to state stability: Basic concepts and results," in *Nonlinear and Optimal Control Theory*, pp. 163–220, Springer, 2006.

[88] S. N. Dashkovskiy, D. V. Efimov, and E. D. Sontag, "Input to state stability and allied system properties," *Automation and Remote Control*, vol. 72, no. 8, pp. 1579–1614, 2011.

[89] N. Rezazadeh and S. S. Kia, "Privacy preservation in a continuous-time static average consensus algorithm over directed graphs," in *American Control Conference*, 2018. to appear.

[90] W. Lohmiller and J.-J. E. Slotine, "On contraction analysis for non-linear systems," *Automatica*, vol. 34, no. 6, pp. 683–696, 1998.

[91] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[92] N. Boffi, S. Tu, N. Matni, J. Slotine, and V. Sindhwani, "Learning stability certificates from data," *arXiv preprint arXiv:2008.05952*, 2020.

[93] H. Tsukamoto, S. Chung, and J. Slotine, "Contraction theory for nonlinear stability analysis and learning-based control: A tutorial overview," *Annual Reviews in Control*, 2021.

[94] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[95] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *arXiv preprint arXiv:2006.04779*, 2020.

[96] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14129–14142, 2020.

[97] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[98] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, *et al.*, "Model-based reinforcement learning for atari," *arXiv preprint arXiv:1903.00374*, 2019.

[99] T. Moerland, J. Broekens, and M. Jonker, "Model-based reinforcement learning: A survey," *arXiv preprint arXiv:2006.16712*, 2020.

[100] D. Sun, S. Jha, and C. Fan, "Learning certified control using contraction metric," *arXiv preprint arXiv:2011.12569*, 2020.

[101] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *arXiv preprint arXiv:1705.08551*, 2017.

[102] S. Vaskov, S. Kousik, H. Larson, F. Bu, J. Ward, S. Worrall, M. Johnson-Roberson, and R. Vasudevan, "Towards provably not-at-fault control of autonomous robots in arbitrary dynamic environments," *arXiv preprint arXiv:1902.02851*, 2019.

[103] R. Tedrake, "Lqr-trees: Feedback motion planning on sparse randomized trees," 2009.

[104] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[105] S. Herbert, M. Chen, S. Han, S. Bansal, J. Fisac, and C. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 1517–1522, IEEE, 2017.

[106] S. Bansal, M. Chen, F. J.F, and C. Tomlin, "Safe sequential path planning of multi-vehicle systems under presence of disturbances and imperfect information," in *American Control Conference*, 2017.

[107] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*, vol. 3. Prentice hall Upper Saddle River, NJ, 2002.

[108] J. Choi, F. Castañeda, C. Tomlin, and K. Sreenath, "Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions," in *Robotics: Science and Systems*, 2020.

[109] A. Taylor, A. Singletary, Y. Yue, and A. Ames, "Learning for safety-critical control with control barrier functions," *Proceedings of Machine Learning Research*, vol. 1, p. 12, 2020.

[110] A. Zaki, A. El-Nagar, M. El-Bardini, and F. Soliman, "Deep learning controller for nonlinear system based on lyapunov stability criterion," *Neural Computing and Applications*, vol. 33, no. 5, pp. 1515–1531, 2021.

[111] S. Richards, F. Berkenkamp, and A. Krause, "The lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems," in *Conference on Robot Learning*, pp. 466–476, PMLR, 2018.

[112] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *IEEE Conference on Decision and Control*, pp. 3717–3724, 2020.

[113] S. Chen, M. Fazlyab, M. Morari, G. Pappas, and V. Preciado, "Learning lyapunov functions for hybrid systems," in *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, pp. 1–11, 2021.

[114] H. Tsukamoto and S.-J. Chung, "Learning-based robust motion planning with guaranteed stability: A contraction theory approach," *IEEE Robotics and Automation Letters*, 2021.

[115] S. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[116] J. Umlauft and S. Hirche, "Learning stable stochastic nonlinear dynamical systems," in *International Conference on Machine Learning*, pp. 3502–3510, PMLR, 2017.

[117] S. Singh, V. Sindhwani, J. Slotine, and M. Pavone, "Learning stabilizable dynamical systems via control contraction metrics," *arXiv preprint arXiv:1808.00113*, 2018.

[118] A. Taylor, V. Dorobantu, H. Le, Y. Yue, and A. Ames, "Episodic learning with control lyapunov functions for uncertain robotic systems," in *International Conference on Intelligent Robots and Systems*, pp. 6878–6884, IEEE, 2019.

[119] J. Kolter and G. Manek, "Learning stable deep dynamics models," *Advances in Neural Information Processing Systems*, vol. 32, pp. 11128–11136, 2019.

[120] F. Bullo, J. Cortés, and S. Martínez, *Distributed Control of Robotic Networks*. Applied Mathematics Series, Princeton University Press, 2009.

[121] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The Collected Works of Wassily Hoeffding*, pp. 409–426, Springer, 1994.

[122] W. Lohmiller and J.-J. E. J. J. E. Slotine, "On contraction analysis for non-linear systems," *Automatica*, vol. 34, no. 6, pp. 683–696, 1998.

[123] L. Lovász, "Submodular functions and convexity," in *Mathematical Programming The State of the Art*, pp. 235–257, Springer, 1983.

[124] T. Thomas, C. Leiserson, R. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[125] F. Duchoň, A. .Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014.

[126] E. Garcia, D. Prett, and M. Morari, "Model predictive control: theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[127] E. Lawler, J. Lenstra, A. H. R. Kan, and D. Shmoys, *The traveling salesman problem; a guided tour of combinatorial optimization*. Wiley, Chichester, 1985.

[128] F. Rubin, "A search procedure for Hamilton paths and circuits," *Journal of the ACM*, vol. 21, no. 4, pp. 576–580, 1974.

[129] B. Gharesifard and S. Smith, "Distributed submodular maximization with limited information," *IEEE Tran. on Control of Network Systems*, vol. 5, no. 4, pp. 1635–1645, 2018.

[130] J. Bondy, U. Murty, *et al.*, *Graph theory with applications*, vol. 290. Macmillan London, 1976.

[131] N. Rezazadeh and S. S. Kia, "A sub-modular receding horizon approach to persistent monitoring for a group of mobile agents over an urban area," *IFAC-PapersOnLine*, vol. 52, no. 20, pp. 217–222, 2019.

[132] N. Rezazadeh and S. S. Kia, "A sub-modular receding horizon solution for mobile multi-agent persistent monitoring: animated numerical example," 2019. https://youtu.be/8NE28UjyLOQ.

[133] R. Gomes and A. Krause, "Budgeted nonparametric learning from data streams," in *ICML*, 2010.

[134] P. Flandrin, *Explorations in time-frequency analysis*. Cambridge University Press, 2018.

[135] R. Hermann and A. J. Krener, "Nonlinear controllability and observability," *IEEE Tran. on Automatic Control*, no. 5, pp. 728–740, 1977.

[136] E. D. Sontag, *Mathematical control theory: deterministic finite dimensional systems*. Springer Science & Business Media, 2013.

[137] M. Hou and R. J. Patton, "Input observability and input reconstruction," *Automatica*, vol. 34, no. 6, pp. 789–794, 1998.

[138] M. L. J. Hautus, "Strong detectability and observers," *Linear Algebra and its Applications*, vol. 50, pp. 353–368, 1983.

[139] C. Read and R. Wilson, *An atlas of graphs*. Oxford University Press, 2005.

[140] D. I. Ridgley, R. A. Freeman, and K. M. Lynch, "Simple, private, and accurate distributed averaging," in *Allerton Conf. on Communications, Control and Computing*, pp. 446–452, 2019.

[141] Y. Xiong and Z. Li, "Privacy preserving average consensus by adding edge-based perturbation signals," in *Conference on Control Technology and Applications*, pp. 712–717, 2020.

[142] S. Zhang, T. O. Timoudas, and M. A. Dahleh, "Consensus with preserved privacy against neighbor collusion," *Control Theory and Technology*, vol. 18, no. 4, pp. 409–418, 2020.

[143] I. L. D. Ridgley, R. A. Freeman, and K. M. Lynch, "Private and hot-pluggable distributed averaging," *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 988–993, 2020.

[144] D. Scherfgen, "Integral Calculator." `https://www.integral-calculator.com`, 2020.

[145] G. Wood and B. Zhang, "Estimation of the lipschitz constant of a function," *Journal of Global Optimization*, vol. 8, no. 1, pp. 91–103, 1996.

[146] C. Knuth, G. Chou, N. Ozay, and D. Berenson, "Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5129–5136, 2021.

[147] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, 2012.

[148] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *CoRR*, vol. abs/2006.04779, 2020.

[149] S. Singh, A. Majumdar, J.-J. Slotine, and M. Pavone, "Robust online motion planning via contraction theory and convex optimization," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5883–5890, 2017.

[150] A. Venkatraman, M. Hebert, and J. Bagnell, "Improving multi-step prediction of learned time series models," in *AAAI*, 2015.

[151] T. Stachowiak and T. Okada, "A numerical analysis of chaos in the double pendulum," *Chaos, Solitons & Fractals*, vol. 29, no. 2, pp. 417–422, 2006.