UNIVERSITY OF SOUTHAMPTON
Faculty of Physical and Applied Sciences
Electronics and Computer Science

Budget–Limited Multi–Armed Bandits

by Long Tran–Thanh

Supervisors: Nicholas R. Jennings and Alex Rogers
Examiners: Nicolò Cesa-Bianchi and Jörg Fliege

A thesis submitted in partial fulfilment for the
degree of Doctor of Philosophy

April 2012

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

Faculty of Physical and Applied Sciences
ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Long Tran–Thanh

Decision making under uncertainty is one of the most important challenges within the research field of artificial intelligence, as they present many everyday situations that agents have to face. Within these situations, an agent has to choose from a set of options, whose payoff is uncertain (i.e. unknown and nondeterministic) to the agent. Common to such decision making problems is the need of balancing between exploration and exploitation, where the agent, in order to maximise its total payoff, must decide whether to choose the option expected to provide the best payoff (exploitation) or to try an alternative option for potential future benefit (exploration).

Among many decision under uncertainty abstractions, multi–armed bandits are perhaps one of the most common and best studied, as they present one of the clearest examples of the trade–off between exploration and exploitation. Whilst the standard bandit model has a broad applicability, it does not completely describe a number of real–world decision making problems. Specifically, in many cases, pulling choice of arm (i.e. making a decision) is further constrained by several costs or limitations. In this thesis, we introduce the budget–limited bandit model, a variant of the standard bandits, in which pulling an arm is costly, and is limited by a fixed budget. This model is motivated by a number of real–world applications, such as wireless sensor networks, or online advertisement. We demonstrate that our bandit model cannot be reduced to other existing bandits, as it requires a different optimal behaviour. Given this, the main objective of this thesis is to provide novel pulling algorithms that efficiently tackle the budget–limited bandit problem. Such algorithms, however, have to meet a number of requirements from both the empirical and the theoretical perspectives. The former refers to the constraints desired by the motivations of real–world applications, whilst the latter aims to provide theoretical performance guarantees.

To begin with, we propose a simple pulling algorithm, the budget–limited ε–first, that addresses the empirical requirements. In more detail, the budget–limited ε–first algorithm is an empirically efficient algorithm with low computational cost, which, however, does not fulfil the theoretical requirements. To provide theoretical guarantees, we introduce two budget–limited UCB based algorithms, namely: KUBE and fractional KUBE,

that efficiently tackle the theoretical requirements. In particular, we prove that these algorithms achieve asymptotically optimal performance regret bounds, which only differ from the best optimal bound by a constant factor. However, we demonstrate in extensive simulations that these algorithms are typically outperformed by the budget–limited ε–first. As a result, to efficiently trade off between theoretical and empirical requirements, we develop two decreasing ε–greedy based approaches, namely: KDE and fractional KDE, that achieve good performance from both the theoretical and the empirical perspective. Specifically, we show that, similar to the budget–limited UCB based algorithms, both KDE and fractional KDE achieve asymptotically optimal performance regret bounds. In addition, we also demonstrate that these algorithms perform well, compared to the budget–limited ε–first.

To provide a grounding for the algorithms we develop, the second part of this thesis contains a running example of a wireless sensor network (WSN) scenario, in which we tackle the problem of long–term information collection, a key research challenge within the domain of WSNs. In more detail, we demonstrate that by using the budget–limited bandit algorithms, we advance the state–of–the–art within this domain. In so doing, we first decompose the problem of long–term information collection into two sub–problems, namely the energy management and the maximal information throughput routing problems. We then tackle the former with a budget–limited multi–armed bandit based approach, and we propose an optimal decentralised algorithm for the latter. Following this, we demonstrate that the budget–limited bandit based energy management, in conjunction with the optimal routing algorithm, outperforms the state–of–the–art information collecting algorithms in the domain of WSNs.

Contents

List of Figures

List of Tables

List of Algorithms

Declaration of Authorship

I, Long Tran-Thanh, declare that the thesis entitled *Budget–Limited Multi–Armed Bandits* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published in a number of conference and journal papers (see Section 1.3 for a detailed list).

Signed:.....

Date:.....

Acknowledgements

Whilst writing the last sentences of the thesis, I realise that pursuing a PhD degree was indeed a long, but very fascinating journey. It involves lots of hard work, intense focus, and self–motivation. But it also brings joy and fun, especially when the light at the end of the tunnel becomes visible. Therefore, I am incredibly thankful to all of the people who encouraged and supported me to keep going on this road.

First and foremost, I would like to thank my supervisors Nick Jennings and Alex Rogers, for their invaluable support and guidance throughout this process. They have become my role models of research during the years at the School of Electronics and Computer Science at University of Southampton, as they always put high standards on my work, teaching me not to be satisfied with half–ready ideas. I am also incredibly thankful for their time and patience in correcting my papers, and for their enlightening discussions that brought me closer to the solution. The time and effort, that they have invested in me, indeed really helped me on my path towards becoming a researcher.

To my co–authors, with whom I have spent wonderful times of (not only) working: Archie Chapman for the great help in the topic of multi–armed bandits, Johnsen Kho for the guidance into the field of wireless sensor networks, and Maria Polukarov for introducing me the beauty of algorithmic game theory and congestion games (which unfortunately is excluded from this thesis). Their advices have been an important influence on my work.

To my Hungarian mentors Janos Levendovszky and Tien Van Do. Tien opened the door of science for me when he invited me to join his research lab in my first year at the Budapest University of Technology and Economics. Later, Janos became my advisor, and has showed me the beauty of Maths.

To my colleagues and friends at the Intelligence, Agents, Multimedia (now Agents, Interactions, Complexity) research group, for their friendship and for providing unforgettable moments: Seb, Rama, Henry Cuong, Dong, Enrico, Valentin, Maria, Francesco, Ruben, Muddasser, Oli, Victor, Tom, Lampros, Matteo, Sasha, Colin, Kate, James, Sid, Sam, Till, Simon, Archie, Enrique, and Thanasis.

To Csaba Szepesvári, for the useful advices and suggestions that brought me closer to the solutions.

To my dear friends, Attila Körösi and Gergely Kiss, for spending their time to proofread my long and dull mathematical proofs.

To the Vietnamese Society at University of Southampton, especially to my flatmates, who made Southampton my second home.

To my parents and my sister, for their unconditional support, love and encouragement. Last but not least, to my love, Trang, who was my inspiration that gave me strength and faith in challenging times of my study. This thesis would never have been written without their love and moral support. To them I dedicate this thesis.

Nomenclature

General

A	Pulling algorithm
A^*	Optimal pulling algorithm
B	Budget limit of the multi–armed bandit
B_t	Residual budget limit at time step t
$G^B(A)$	Total received reward of algorithm A with respect to budget limit B
I^*	Arm with highest expected reward density
K	Number of arms within a multi–armed bandit
$M^*(B_t)$	best estimated combination of arms with respect to B_t
$N^B(A)$	Number of times algorithm A pulls arm i with respect to budget limit B
$R^B(A)$	Total regret of algorithm A with respect to budget limit B
$I^+(t)$	Arm with highest estimated expected reward density at time step t
$\hat{I}(t)$	Arm with highest estimated upper confidence bound density at time step t
T	Total number of pulls within the multi–armed bandit
c_i	Pulling cost of arm i
$c_{\min}$	Minimal pulling cost
$c_{\max}$	Maximal pulling cost
d_i	The difference between the reward density of arm i at that of I^*
$d_{\min}$	The lowest value of d_i among all $d_i > 0$
$i(t)$	Arm pulled at time step t
$m^*_{i,t}$	number of times arm i is included in the best combination $M^*(B_t)$
$\{m_{i,t}\}$	Combination of arms at time step t
$n_{i,t}$	Number of pulls of arm i until time step t
$p_i(t)$	Probability of pulling arm i at time step t
$r(t)$	Reward value received at time step t
t	time step
Δ_i	The difference between the expected reward of arm I^* at that of arm i
δ_i	The difference between the cost of arm i at that of I^*

μ_i	Expected reward value of arm i
μ^*	Best Expected reward value
$\hat{\mu}_{i,n_{i,t}}$	Estimated value of arm i's expected reward at time step t, after $n_{i,t}$ pulls

Chapter 2

$I(t)$	Arm with highest estimated expected reward value
L	limit of number of pulls per time step in Comband
$R^T(A)$	Regret value of algorithm A until time step T
r_t	Single regret value at time step t
$r_i(t)$	Reward value received at time step t by pulling arm i
$w_i(t)$	Weight value of arm i at time step t in Exp3
$\hat{\mu}_{i,t}$	Estimated value of arm i's expected reward at time step t
τ_i	Starting time of τ^{th} epoch of arm i in UCB2

Chapter 4

A_{uniform}	Uniform pulling algorithm within the exploration phase
A_{greedy}	Density–ordered greedy based exploitation algorithm
I^+	Arm with highest estimated expected reward density after exploration
$I^{\min}$	Arm with lowest reward density
N_i	Number of times arm i is pulled in A^*
$d_{\max}$	The largest value of d_i
n_i	Number of times arm i is pulled within the exploration phase
β	Probably approximately correct (PAC) learning factor
ε	Tuning parameter for exploration
$\hat{\mu}_{i,n_i}$	Estimated value of arm i's expected reward after exploration

Chapter 5

$N_j(T)$	number of times arm i is pulled with respect to T
$b_{t,s}$	Confidence bound at time step t, after s pulls

Chapter 6

$n_{j,t}^{\mathrm{R}}$	Number of times arm i is randomly chosen from the uniform distribution to pull until time step t
γ	Tuning factor
ε_t	Exploration factor at time step t

Chapter 7

$\mathfrak{A}_i$	set of energy budget allocation combinations of agent i
B_i	Initial battery capacity of agent i
$B_i(t)$	Residual battery capacity of agent i at time step t
$B_i^{\mathrm{Rx}}(t)$	Energy budget that agent i allocates to receiving
$B_i^{\mathrm{S}}(t)$	Energy budget that agent i allocates to sampling
$B_i^{\mathrm{Tx}}(t)$	Energy budget that agent i allocates to transmission
BS	Base station
$D_{r_j}(\tau)$	Maximal number of packets that agent i can receive at communication round τ
I	Set of agents
L	Number of layers
$\mathfrak{L}_l$	Set of agents in layer l
N_i^{Rx}	Receiving capacity of agent i at each time step
N_i^{S}	Sampling capacity of agent i at each time step
N_i^{Tx}	Transmission capacity of agent i at each time step
$\mathbf{Q}_i(t)$	Set of packets within the queue to send of agent i at time step t
$\mathbf{Re}_i(t)$	Set of residual packets within the memory of agent i at time step t
$\mathbf{Rx}_i(t)$	Set of packets that agent i receives at at time step t
$\mathbf{S}_i(t)$	Set of packets that agent i samples at at time step t
$\mathbf{Tx}_i(t)$	Set of packets that agent i transmits at at time step t
T	Total operating time
d_i	Distance from BS (in hops)
e_i^{Rx}	Energy consumption of receiving a single unit
e_i^{S}	Energy consumption of sampling a single unit
e_i^{Tx}	Energy consumption of transmitting a single unit
p	Data packet
$r_i(t)$	Reward value of agent i at time step t
r_j	Agent from a receiver layer
s_i	Agent from a sender layer
$v(p,t)$	information value of packet p at time step t
t	time step

λ	Information durability factor
τ	Communication round within a single time step

Chapter 1

Introduction

In many everyday situations, an *agent*, or a decision maker, has to choose between alternatives in order to achieve its goal. These situations vary from simple daily routines, such as driving a car to work or food to buy, to complex and important problems, such as the design of clinical trials or financial investments. In particular, in the former, a driver aims to arrive to her workplace on time, and thus, everyday she chooses a driving route that she believes to be the fastest. On the other hand, the latter consists of experiments that aim to determine which medicines offer the best treatment for patients with a certain disease. Now, the ingredient that makes such decision making difficult is the *uncertainty* in the outcome of the decision. More precisely, the outcome of the decision, which is typically a *reward*, or a cost, is only revealed to the agent after the decision is made. Furthermore, this outcome is typically affected by other things, whose effects are not known to the agent, and thus, it may be uncertain. For example, on any particular day the driver does not know beforehand what the traffic on the chosen route will be, and the success or failure of the chosen treatment is not guaranteed for any particular patient. Given this, in order to maximise its performance (i.e. *exploitation*), the agent has to gather information that improves knowledge of the environment by trying out different alternative decisions (i.e. *exploration*). Exploitation and exploration decisions, however, have to be carefully made. If the agent focuses solely on exploration, it will gain accurate information about the environment, but might not be able to maximise its total reward. On the other hand, by putting more effort on exploiting, the agent might miss a chance to find a better alternative. Thus, one of the most crucial challenges in decision–making under uncertainty is the problem of finding a *trade–off between exploration and exploitation*.

One of the clearest examples of this trade–off is presented in the standard, or stochastic, *multi–armed bandit* (MAB) problem, originally proposed by Robbins (1952). The term "bandit" refers to the usual name of a gambling slot machine ("one–armed bandit") which has one arm which can be pulled. The standard MAB problem is a generalisation

of this one–armed bandit, which consists of a single machine with K arms, each of which delivers rewards that are *independently drawn from unknown distributions* when each arm is pulled. Given this, an agent must choose which of these arms to pull. At each time step, it pulls one of the machine's arms and receives a reward. The agent's goal is to maximise the expected sum of the rewards it receives over a sequence of pulls. If the distributions were known, this goal would be equivalent to finding the arm with the highest expected payoff, and then to keep playing using that best arm. However, the agent does not know the rewards for the arms, so it must sample them in order to learn which is the optimal one. In other words, in order to maximise the total reward (i.e. exploitation) the agent first has to estimate the mean rewards of all of the arms (i.e. exploration).

In the standard MAB, this trade–off has been widely studied from both theoretical and empirical aspects (Agrawal, 1995b; Anderson, 2001; Auer *et al.*, 2002; Lai and Robbins, 1985; Vermorel and Mohri, 2005) . In more detail, in the bandit settings, pulling strategies are referred to as *policies*, and they vary from simple algorithms, such as ε–first (Even-Dar *et al.*, 2002) or ε–greedy (Watkins, 1989), to more advanced methods that use more complex rules to determine the next arm to pull, such as *decreasing ε–greedy* (Auer *et al.*, 2002), POKER (Vermorel and Mohri, 2005) or *upper confidence bound* (UCB) (Auer *et al.*, 2002) [1]. Now, the performance of these policies is often measured in terms of cumulative *regret*, or total loss, which is the difference between the total reward that the policy can achieve, and the total reward received if the theoretical optimal pulling policy (i.e. the policy that maximises the total received reward) is followed. From the theoretical aspect, advanced policies (e.g. UCB–based, or POKER) typically outperform simple methods. In particular, decreasing ε–greedy, POKER, and UCB–based policies achieve *zero–regret*; that is, their average regret (i.e. cumulative regret divided by the number of time steps) converges to 0 with probability 1 as the number of steps tends to infinity. Intuitively, zero–regret policies guarantee *optimal asymptotic convergence*. They converge to an optimal policy as time goes by. This guarantee of asymptotic convergence, however, cannot be achieved with ε–first or ε–greedy.

However, since many real–world applications have a finite operating time interval, asymptotic convergence is typically not sufficient, since it only guarantees convergence as time goes to infinity. Therefore, in addition to asymptotic convergence, there is also a need to provide *regret bounds over finite time*, that uniformly guarantees for every time step that the regret of the policy does not exceed a certain threshold (i.e. the performance of the policy stays close to that of an optimal policy after each time step). In this sense, UCB and decreasing ε–greedy outperform POKER, since they guarantee an efficient regret bound, while POKER does not (for more details, see Chapter 2).

[1] The details of these strategies are given in Section 2.2.

Although theoretical results indicate the dominance of decreasing ε–greedy and UCB–based policies, empirical experiments show that in many real–world applications, ε–first typically outperforms the more advanced policies, even if it cannot guarantee theoretical efficiency (Kuleshov and Precup, 2010; Vermorel and Mohri, 2005). This is especially true when the bandit size is large; i.e. when the bandit problem contains hundreds or thousands of arms (or even more), as is the case in many real–world scenarios (see Chapter 7 for more details). One possible reason is that the constant factor within the theoretical bounds depends on the number of arms, and thus, it is large if the bandit size is large. Given this, besides decreasing ε–greedy and UCB–based policies, the ε–first policy is also widely used in order to tackle the standard bandit problem.

While this standard model has a broad applicability, it does not completely describe a number of real–world sequential decision–making problems. Specifically, in many cases, pulling choice of arm is further constrained by several costs or limitations. These include switching costs (where switching between arms is costly), pulling costs (where pulling arms is costly), limitation of an arm's existence (where arms have a limited life span), or limitation in varying between arms (when the number of changes between arms is limited). Accordingly, recent studies have introduced a variety of related models in order to adapt to these bandit problems (Chakrabarti *et al.*, 2008; Cicirello and Smith, 2005; Guha and Munagala, 2009; Langford and Zhang, 2007), and in particular, a number of researchers have focused on MABs with budget constraints, where arm–pulling is costly and is limited by a fixed budget (Antos *et al.*, 2008; Bubeck *et al.*, 2009; Guha and Munagala, 2007; Madani *et al.*, 2004). In particular, these bandit models include those with a budget limited exploration phase, and a cost–free exploitation phase. This is motivated by a variety of applications. For example, in the shortest driving path scenario, the cost of the fuel consumed by the car differs as the driver choose different routes, or the medical treatment of a particular patient implies a certain financial cost. In both cases, the cost of making a decision (i.e. pulling an arm) can be expressed in terms of money, and the agent is not allowed to exceed a certain limit of expense. Within these scenarios, the agent's goal is to determine the best arm (i.e. decision), but its exploration budget limits the number of times it can sample the arms in order to estimate their rewards, which defines an initial exploration phase. In the subsequent cost–free exploitation phase, an agent's policy is then simply to pull the arm with the highest expected reward.

However, in many scenarios, it is not only the exploration phase, but also the exploitation phase, that is limited by a cost budget. This type of limitation is again well motivated by several real–world applications. For example, consider a company that advertises itself online. It has a limited budget for renting online advertising banners on any of a number of web sites, each of which charges a different rental price. The company wishes to maximise the number of total clicks on its banners, but it does not know the

click–through rate for banners on each site. As such the company needs to estimate the click–through rate for each banner (exploration), and then to choose the combination of banners that maximises the sum of clicks (exploitation). In terms of the model described above, the price of renting an advertising banner from a website is the pulling cost of an arm, and the click–through rate of a banner on a particular website is the true reward for pulling that arm, which is unknown at the outset of the problem. It is obvious that both the exploration and exploitation phases are budget limited within this example.

Another example comes from the domain of wireless sensor networks. In particular, in many such applications, sensor nodes are deployed for collecting information over a prolonged period of time. However, a node's actions (such as sampling or data forwarding) consume energy, and furthermore, it is typically physically infeasible to replace the battery of a particular sensor. Given this, the total number of actions that a single sensor node may make is limited by the capacity of the sensor's batteries. Now, typical sensor network deployments require that sensors learn the optimal combination of actions that can be performed, with the goal of *maximising the collected information over a long term*. Thus, each action can be considered as an arm, with a cost equal to the amount of energy needed to perform that task. Given this, in order to exploit (i.e. take the optimal actions given reward estimates), the sensor has to efficiently explore (i.e. learn the rewards of the tasks), within the battery limit. Now, in these examples, because the total budget (e.g. the research budget, or the advertising budget) is limited, both exploration and exploitation phases are limited as well.

To address this limitation, within this thesis, we introduce a new bandit model. We call this the *budget–limited* MAB, in which pulling an arm is again costly, but crucially *both* the exploration and exploitation phases are limited by a single budget. Note that in this case, if the expected rewards for pulling the arms are known, then the optimal solution is not to repeatedly pull the optimal arm *ad infinitum*, as is in other MAB problems, but rather to pull a *finite combination of arms* that maximises the reward and fully exploits the budget, since a budget–limited MAB can be reduced to an unbounded knapsack problem (Andonov *et al.*, 2000). To see this, consider that pulling an arm corresponds to placing an item into the knapsack, with the arm's expected reward equal to the item's value and the pulling cost the item's weight. The total budget is then the weight capacity of the knapsack. Given this, the optimal combination of items for the knapsack problem is also the optimal combination of pulls for the budget–limited MAB. This difference in desired optimal solution from existing MAB problems means that, when defining a decision–making policy for our problem, we must be cognizant of the fact that an optimal policy will involve pulling a combination of arms. As such, it is not sufficient to learn the expected reward of only the highest–value arm; we must also learn the other arms' rewards, because they may appear in the optimal combination. Importantly, we cannot simply import existing bandit policies, because they concentrate on learning only

the value of the highest expected reward arm, and so will not work in this setting. For example, consider a three–armed bandit, with arms X, Y and Z that have true expected reward and pulling cost values of $(80, 52)$, $(60, 40)$ and $(50, 31)$. Suppose the budget is 185. At this point, the optimal solution is to pull arms X and Y one time each, and Z three times, giving an expected total reward of 290. However, by just focusing on the arm with highest expected reward, as existing bandit algorithms typically do, the resulting total reward is 240 (by pulling arm X three times). In addition, focusing on the arm with the highest reward–cost ratio, a straightforward modification of standard MAB policies to the budget–limited version, is not optimal either. Indeed, Z is the arm highest reward–cost ratio, and by repeatedly pulling Z, the maximal total reward we can get is 250 (by pulling it three times).

It is clear from the abovementioned example that existing bandit algorithms may not be suitable for tackling the budget–limited MAB. Thus, new techniques must be developed for this new problem, which do consider the combinatorial aspect of the optimal solution to the budget–limited MAB problem. To date, however, none of the previous work addresses these issues within the budget–limited MAB (see Chapter 2 for more details). Thus, this thesis seeks to start addressing this gap. Specifically, in Section 1.1, we describe the research requirements that arm pulling algorithms should satisfy, in order to achieve efficient total payoff, with respect to a given budget. We then introduce an application scenario for the budget–limited multi–armed bandits, namely the aforementioned problem of *long–term information collection in wireless sensor networks* in Section 1.2. This scenario will be used as an application environment in which we will evaluate the performance of our proposed budget–limited MAB algorithms. Following that, we introduce the contributions of this thesis in Section 1.3. Finally, Section 1.4 outlines the overall structure of the remainder of this thesis.

1.1 Research Requirements

The aim of the work in this thesis is to design pulling policies that maximise the total reward, with respect to the overall budget limit. Such policies, however, have to meet a number of requirements in order to achieve the aforementioned goal. In particular, research requirements for a pulling policy can be divided into *empirical* and *theoretical* requirements. The former refers to the constraints desired by real–world applications, while the latter aims to provide theoretical performance guarantees. Note that empirical requirements typically focus on guaranteeing the good performance of the algorithm in average situations, while theoretical requirements guarantee good performance even for the worse case. Given this, it might occur that an algorithm with good empirical performance may fail in extreme (i.e. worse case) situations. In contrast, an algorithm with theoretical guarantees may be outperformed by other, theoretically well founded,

algorithms, as is the case within the multi–armed bandits (see 2.2 for more details). As a result, we consider both types of requirements in this thesis. The broad empirical requirements that a pulling policy should satisfy are the following:

1. **Experimental performance quality (Requirement 1)**: Since the budget–limited MAB is motivated by many real–world applications, it is important to design policies that achieve high performance quality (i.e. low regret) within real–world settings. In particular, real–world applications typically have large problem size (i.e. the number of arms is high). Given this, a pulling policy has to be able to efficiently deal with this large problem size, and provide high performance quality.

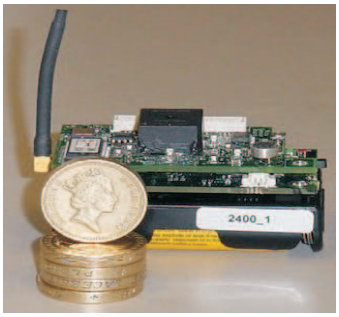
2. **Computational feasibility (Requirement 2)**: In many cases, the agent has to make quick decisions, that have to be calculated in a short period of time. In addition, many real–world applications have low computational capacity as well. For example, wireless sensor nodes are limited in memory and computational capability. Given this, they are not suitable for computationally expensive methods. Consequently, it is necessary to develop efficient policies that have low computational cost.

Apart from these empirical requirements, we also mentioned our theoretical research aim in the discussion above, namely *efficient finite–time regret bound*:

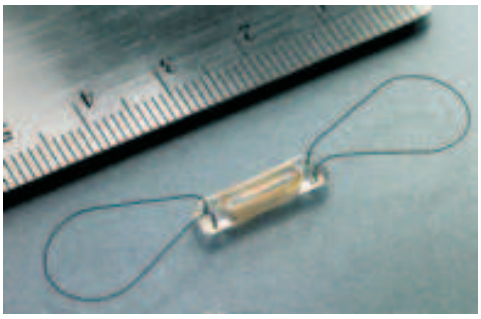
3. **Efficient finite–time regret bound (Requirement 3)**: Due to the finite overall budget, budget–limited MAB policies have to operate over a finite time interval. Thus, it is important to guarantee that for any budget size, the regret is bounded. That is, the performance of the proposed policy has to be efficient so that it is always close to that of the optimal solution. In addition, a pulling policy should be able to learn the optimal solution in the long term. Given this, it is desirable to have policies that converge to the optimal policy with probability 1 as the budget tends towards infinity (i.e. there is enough budget to learn the optimal behaviour).

1.2 Application Scenario

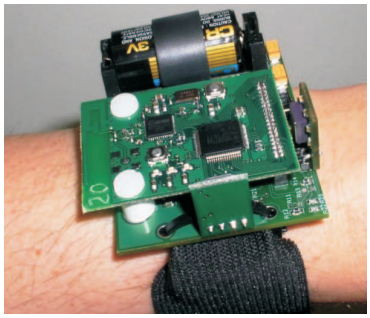
Given the research requirements above, we now consider how we can efficiently use the budget–limited MAB model to tackle real–world challenges. In so doing, we first describe the problem of long–term information collection in wireless sensor networks, which is one of the real–world motivations of the budget–limited MAB model. In particular, efficient long–term information collection is a key challenge within the domain of wireless sensor networks (Rogers *et al.*, 2009; Stankovic, 2004), and is gaining attention of a large number of research studies (Dekorsy *et al.*, 2007; Kho *et al.*, 2010; Merrett, 2008; Ok



a) wireless sensor mote[1]



b) wireless cardio sensor[2]



c) wearable sensor[3]

d) robot football[4]

FIGURE 1.1: Wireless sensor nodes.

et al., 2009). We then show how the aforementioned research requirements have to be addressed in this scenario, and we also identify additional design requirements that we have to take into account.

In more detail, wireless sensor networks (WSNs) are now being increasingly used in a wide variety of applications, ranging from environmental, habitat and traffic monitoring, to object tracking and military field observations (Rogers *et al.*, 2009; Romer and Mattern, 2004). In WSNs, each wireless sensor node is typically equipped with a sensing module for sensing data from the surrounding environment, a radio transceiver module for wireless communication, a small microcontroller as the processing unit, an external memory for data storage and a limited energy source (usually a battery). The size of a single sensor node can vary from the size of a shoe box to the size of a coin (see Figure 1.1). Furthermore, their cost is similarly variable, ranging from hundreds of pounds to a few pence, constrained by parameters such as size, energy, memory, computational speed and communication bandwidth required of individual nodes (Romer and Mattern, 2004). These networks are typically deployed for collecting information from the environment, which is then forwarded in data packets to a base station (BS), for further

[1]Taken from link `http://www.npl.co.uk/server.php?show=ConWebDoc.289`;
[2]Taken from link `http://amp.osu.edu/news/article.cfm?ID=4576`;
[3]Taken from link `http://www.intelligent-systems.info/biofeedback/biofeedback.htm`;
[4]Taken from link `http://amp.osu.edu/news/article.cfm?ID=4576`.
All links are checked on 08/12/2011.

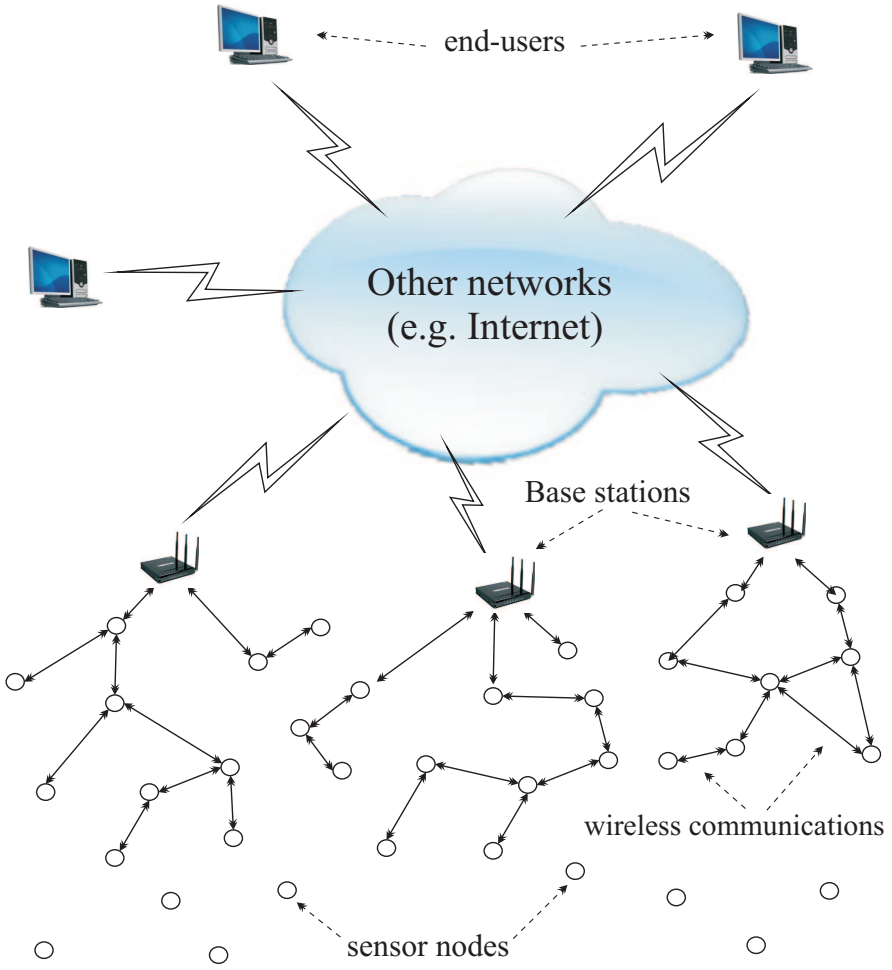


FIGURE 1.2: Typical wireless sensor network.

processing. Such systems are typically required to operate over an extended period of time (covering months or even years). Figure 1.2 depicts a typical topology of WSNs.

Note that some real–world WSNs requires newest data only, and thus, the value of information that is sampled in the past rapidly decays as time passes by. Such WSNs are typically deployed for real–time target tracking or real–time object localisation (He *et al.*, 2006; Simon *et al.*, 2004). Within these networks, a fundamental goal is to send collected data to the BS as fast as possible (i.e. the data has a strict delivery time constraint). On the other hand, other networks focus on collecting information within a non real–time manner. That is, the deployed network continuously collects information from the surrounding environment, without having the aforementioned strict delivery time constraint (i.e. the collected information can be delayed for a longer time before it is delivered to the BS). Since most of the WSN applications are deployed to fulfil the latter type of monitoring (Chong and Kumar, 2003; Merrett, 2008; Rogers *et al.*, 2009), here we focus on networks where the goal is to collect information over a period of time, in a non real–time manner.

Given this, the objective of this scenario is to develop policies that *maximise the amount of information collected by the WSN and delivered to the BS, over a given time interval, in a non real–time manner*. In addition, we want to avoid centralised approaches, as this needs global information in order to achieve maximal data collection, and could represent a significant computational bottleneck (Boukerche, 2008; Wagner and Wattenhofer, 2007). For these reasons, we focus on *decentralised* approaches, in which there is no central unit that coordinates the actions of the individual sensors. This approach, however, leads to several issues. Specifically, to achieve system–wide goals, the nodes must typically coordinate their actions with their neighbours (e.g. to forward data or to track objects). In addition, since the nodes typically operate in a dynamically changing environment, they must be able to *autonomously* adapt their behaviour, without having any global information about the system, in order to achieve long-term global goals (e.g. maximal data collection or optimal coverage). Now, since WSNs are heavily resource constrained (i.e. low energy capacity, size and computational constraints) (Akyildiz *et al.*, 2002; Rogers *et al.*, 2009), this results in a number of significant and specific research issues that have to be addressed. In particular, limited energy capacity demands *energy–awareness*. That is, it is necessary to efficiently manage the energy consumption of the nodes. Otherwise, rapid battery depletion may lead to insufficient data collection from the network.

Against this background, the information collection problem that we address consists of a set of sensor nodes, collecting information from their surrounding environment over an extended period of time, without the aid of a centralised controller. Due to the limited energy capacity of the nodes, energy efficiency is perhaps the most important issue within the information collection problem (Chong and Kumar, 2003; Stankovic, 2004). Given this, it is important to wisely *manage the energy consumption* of the nodes, such that they can decide whether to allocate more of this scarce resource to the tasks of sampling, receiving, or transmitting data, in order to achieve maximal long–term information collection. In addition, we also need to develop *routing techniques* in order to deliver the data to the BS, and thus, to maximise the amount of information collected in the network. Given this, in this scenario, we focus in particular on the challenges of energy management and data routing.

Now, we show how this problem naturally maps onto a budget–limited MAB. In particular, in order to collect information from the environment, the agents can choose a combination of data sampling, receiving, and transmission at each time step. These actions all consume energy, and different combinations (i.e. arms) need different amounts of energy (i.e. pulling cost). Furthermore, since the capacity of the agents' batteries is limited, the total number of actions is limited over time as well, and thus, this limited capacity can be seen as the budget of the bandit model. Note that the goal of the WSN is to maximise the total amount of collected information over a prolonged period

of time. Given this, by considering the amount of collected information at each time step as the reward that the agent gets by choosing a particular combination of sensory actions, we can model the information collection problem that a sensor agent is facing as a budget–limited MAB.

We now return to the importance of the research requirements described in Section 1.1 within this scenario as follows. Note that the set of possible combination of actions, that an agent can choose from, is typically large (see Chapter 7 for more details). Therefore, the proposed approach has to be efficient in terms of tackling the problem of long–term information collection within large MAB models (performance quality). In addition, since the sensor agents are heavily resource constrained, it is important to develop information collecting methods that do not require high computational cost (computational feasibility). Now, since WSNs can be deployed in a large variety of environments (see Chapter 7), high quality empirical results might not be sufficient, since the existence of pathological behaviour cannot be ruled out. Thus, it is important to provide theoretical performance guarantee as well (finite–time regret bounds). Given this, long–term information collection in wireless sensor networks can be seen as a suitable application for our bandit model.

However, beside the abovementioned requirements, efficient mechanisms to maximise long–term information collection in WSNs have to deal with a number of additional issues related to the significant physical constraints, such as node malfunctioning, or limited communication (see Chapter 7 for more details). Given this, to design such mechanisms, we also have to take the following requirements into account:

4. **Adaptivity (Requirement 4)**: Since the sensor agents are typically deployed in *a priori* unknown environments, efficient performance cannot be sustained without the ability to learn and to adapt to the (unknown) environmental characteristics. That is, the nodes should be able to learn efficient policies on–line, based on their own experiences. Moreover, they have to achieve an efficient trade–off between exploration and exploitation.

5. **Robustness and flexibility (Requirement 5)**: Due to the long operating time of the network, node failures and lossy communication links are likely to occur. Even in these cases, the network operation should not collapse, rather it should degrade gracefully. Therefore, the nodes should be able to handle these situations, by ensuring that the operation of the remaining nodes is only minimally affected.

6. **Limited use of communication (Requirement 6)**: Since communication is typically the most expensive task in WSNs, a good information collection approach should avoid having significant communication cost (i.e. the energy amount allocated to sending control messages), in order to achieve efficient performance in

information collection. In particular, by reducing the amount of energy for communication, the sensor agents can allocate more energy to forwarding real data, and thus, it can increase the amount of collected information.

Having explained the context of the research conducted within this thesis, we now detail the specific research contributions.

1.3 Research Contributions

Given the requirements described in Section 1.1, our research aim is to develop pulling policies for efficiently tackling the budget–limited multi–armed bandit problem. In so doing, we contribute to the state–of–the–art and address gradually all four requirements by developing a number of novel classes of pulling policies. More specifically, we make four main contributions in this thesis. The first consists of a simple pulling policy that addresses the empirical requirements, that is, Requirements 1 and 2 (Chapter 4). The second focuses on addressing the theoretical requirement by proposing more advanced policies (Chapter 5). These methods, however, demonstrate poor performance in the scenario of long–term information collection of WSNs (i.e. they fail to fully satisfy the empirical requirements). Against this background, the third group of contributions proposes a trade–off between the two above, and proposes pulling algorithms that perform well from both theoretical and empirical aspects (Chapter 6). Within the last contribution group, we specifically address the research challenges within the problem of long–term information collection in WSNs described in Section 1.2. These contributions are summarised in Table 1.1.

Now, for each contribution, we briefly highlight their most salient properties in terms of the requirements discussed earlier as follows.

1. **Budget–limited ε–first approach (Chapter 4):** The first group of contributions in this thesis addresses Requirements 1 and 2 by introducing the first pulling policy, called the budget–limited ε–first approach, that efficiently tackles the budget–limited MAB problem.

   In particular, the budget–limited ε–first approach splits the total budget B into two portions, the first εB of which is used for exploration, and the remaining $(1-\varepsilon)B$ for exploitation. In the exploration phase, the agent *uniformly* samples the arms (i.e. the arms are sequentially pulled one after the other) to construct estimates of their expected rewards. It then uses these estimates to calculate the optimal combination of pulls to undertake in the exploitation phase. The key benefit of this approach is that we can easily measure the accuracy of the estimates

	Chapter 4	Chapter 5		Chapter 6	
	budget–limited ε–first	KUBE	fractional KUBE	KDE	fractional KDE
Experimental performance quality	++(*)	+	-	++(*)	+
Computational feasibility	++(*)	+	++	+	++
Finite–time regret bound	-	++(*)	++(*)	++	++

TABLE 1.1: An overview of our contributions in terms of the research requirements in the budget–limited MAB domain. The symbols have the following meaning: '+' ('++') means that the requirement is (strongly) satisfied. In addition, '(*)' indicates the best performance of the row. On the other hand, '-' means the requirement is not satisfied.

associated with a particular value of ε, because all of the arms are sampled the same number of times. Hence, we can control the performance regret as a function of ε, which gives us a method of choosing an optimal ε for a given scenario. Given this, our contributions within this chapter can be detailed as follows:

- We show that the computational complexity of the budget–limited ε–first approach is $O(\varepsilon KB + K \ln K)$ at each time step. That is, the policy has low computational cost (Requirement 2).
- We provide a $O(B)$ regret bound for the budget–limited ε–first approach, which fails to fulfil Requirement 3, since it does not guarantee the convergence of the budget–limited ε–first approach to the optimal solution as B tends towards infinity.
- However, we improve the regret bound above by proving that with large probability, the budget–limited ε–first approach can achieve a $O\left(B^{\frac{2}{3}}\right)$ regret bound. That is, Requirement 3 can be partially satisfied (i.e. the budget–limited ε–first approach converges to the optimal policy with high probability).
- We demonstrate that, despite the weak theoretical regret bound, the budget–limited ε–first approach still achieves efficient performance in tackling the problem of long–term information collection within WSNs (Requirement 1).

2. **Budget–limited upper confidence bound based approaches (Chapter 5):** The second group of contributions extends the abovementioned results by addressing Requirement 3 (i.e. efficient finite–time regret bound). In particular, we propose two UCB–like policies, namely: (i) the *knapsack based upper confidence bound exploration and exploitation* (KUBE); and (ii) *fractional* KUBE, the first pulling policies that achieve logarithmic regret bound within the domain of

budget–limited MAB. Unlike the budget–limited ε–first approach, these policies do not explicitly separate exploration from exploitation. Instead, at each time step, they calculate the best combination of arms that provides the *highest total upper confidence bound* of the estimated expected reward, and still fits into the residual budget, using an unbounded knapsack model to determine this best combination (Andonov *et al.*, 2000). Note that the use of these techniques is common in the MAB domain, as they present elegant ways to efficiently tackle the trade–off between exploration and exploitation (Agrawal, 1995b; Audibert *et al.*, 2009; Auer *et al.*, 2002; Auer and Ortner, 2010). Following this, they then use the frequency that each arm occurs within this approximated best combination as a *probability* with which to randomly choose an arm to pull in the next time step. The reward that is received is then used to update the estimate of the pulled arm's expected reward, and the unbounded knapsack problem is solved again.

Now, since unbounded knapsack problems are known to be NP–hard, efficient approximation methods are needed in order to fulfil our empirical research requirements. Given this, KUBE uses an efficient approximation method taken from the knapsack literature, called the *density–ordered greedy* approach, in order to estimate the best combination (Kohli *et al.*, 2004). Conversely, fractional KUBE uses a different approximation approach to tackle the knapsack problem. In particular, it relaxes the unbounded knapsack to a fractional version, where fractions of items are allowed (Kellerer *et al.*, 2004; Marcello and Toth, 1990). Since the fractional version is computationally less expensive than the density–ordered greedy method, fractional KUBE clearly has lower computational cost, compared to that of KUBE. However, this computational gain is balanced by a decreased performance quality. In particular, the specific contributions within this group can be described as follows:

- We show that the computational complexity of KUBE is $O(BK \ln K)$ at each time step, where K is the number of arms. In addition, we also show that fractional KUBE has a decreased computational complexity of $O(BK)$. That is, both KUBE and fractional KUBE are computationally more expensive, compared to the budget–limited ε–first approach, but they still have a polynomial computational complexity (Requirement 2).
- We provide a $O(\ln B)$ upper bound for the performance regret of KUBE and fractional KUBE respectively. This implies that these policies are also a zero–regret policy, and thus, they satisfy Requirement 3.
- We also show that this logarithmic bound is asymptotically optimal. That is, it only differs from the best possible regret bound by a constant factor. However, we demonstrate that the constant factor within the lower bound

of fractional KUBE is larger than that of KUBE. That is, between the two, fractional KUBE has a worse lower bound.

- We demonstrate that KUBE typically outperforms its fractional counterpart in tackling the problem of long–term information collection within WSNs. However, they are both outperformed by the budget–limited ε–first approach in many cases. That is, they both fail to address research Requirement 1 (i.e. experimental performance quality).

3. **Budget–limited decreasing ε–greedy based approaches (Chapter 6):** As discussed above, the third group of contributions is dedicated to determining a trade–off between satisfying both empirical and theoretical requirements. In so doing, we introduce a class of ε–greedy based policies, which consists of two pulling policies. The first is more efficient, but requires a computationally expensive algorithm, namely *knapsack based decreasing ε–greedy* (KDE), while the second one, called *fractional* KDE, is computationally less expensive, but provides weaker performance, compared to that of the former. In more detail, similar to the UCB–based algorithms, KDE and its fractional counterpart also use the unbounded knapsack approach to determine the best combination of arms that provides the *highest total estimated expected reward* at each time step t. Following this, they randomly choose between the probability distribution created from the frequency with which the arms occur in this best combination and the uniform distribution with probability $(1 - \varepsilon_t)$ and ε_t, respectively. From the chosen distribution, the algorithms then randomly draw an arm to pull in the next time step. Again, similarly to the case of UCB–based policies, KDE and its fractional counterpart differs from each other in the way they solve the unbounded knapsack. In particular, KDE uses the density–ordered greedy, while fractional KDE uses the fractional knapsack model. Thus, the contributions related to these policies can be detailed as follows:

   - We show that the computational complexity of KDE and its fractional counterpart is $O(BK \ln K)$ and $O(BK)$, respectively. These results are similar to that of the UCB–based algorithms (Requirement 2).
   - We provide a $O(\ln B)$ upper bound for the performance regret of KDE and fractional KDE. This implies that these policies are asymptotically optimal in terms of minimising the performance regret. Consequently, they satisfy Requirement 3 (i.e. efficient finite–time regret bound). We also show that whereas fractional KDE is computationally more efficient than KDE, it has a worse lower bound and is less efficient within the application scenario described in Section 1.2.
   - We demonstrate that KDE achieves good performance in practice (Requirement 1). In particular, KDE achieves similar performance, compared to that

of the budget–limited ε–first approach. On the other hand, fractional KDE is outperformed by budget–limited ε–first in many cases.

We now turn to the contributions that address the research challenges within the WSN domain discussed in Section 1.2. In particular, as previously discussed, we focus on the challenges of energy management and data routing. However, tackling this joint problem of energy management and routing is hard. In particular, each agent has a number of options to allocate amounts of energy to its sensory tasks. In addition, it needs to decide which packet it has to send, and to whom among its neighbouring agents. These options together result in a large task combination space (i.e. the space of combined tasks of energy allocation and packet transmission/receiving), from which the agent has to determine an optimal one (i.e. the task combination that leads to the desired goal of the network). This task combination space is typically exponential, compared to the size of the network, so the joint problem quickly becomes infeasible in terms of complexity. Thus, to simplify the complexity of the original joint problem, we separate the energy management and data routing problems. However, as we will show, by using the solutions of the separated problems, efficient information collection can be still achieved.

In more detail, the decomposition of the original problem can be described as follows. It is based on the observation that by adaptively setting the value of the energy budgets allocated to the various sensory tasks, the agents should achieve better performance in dynamic environments than systems without the ability to adapt in this fashion. However, in order to determine which energy budget allocation combinations are optimal (exploitation), the agent first has to learn the performance of all the combinations (exploration). Thus, it has to balance between exploration and exploitation. Given this, within the energy management problem, we seek for an efficient learning method that finds a trade–off between exploring and exploiting the energy budget allocation combinations, in order to achieve optimal performance of long–term information collection. Now, suppose that all the agents have already set their energy budget value for sampling, receiving, and transmitting tasks. In this case, to maximise the value of the total collected information, it is obvious that we need to maximise the total information value of data sampled or relayed by agents that are one hop from the BS. The latter, however, is equal to data that is sampled or relayed by agents that are two hops from the BS, and so on. Thus, it is also important to maximise the *information throughput* (i.e. the total transmitted information value) between neighbouring *layers* of agents (i.e. the group of agents that are the same distance from the BS) by using efficient routing techniques. This forms the routing problem we aim to solve within this application scenario. Given this context, this work advances the–state–of–the–art in the following specific ways:

	Chapter 7		
	MAB/EM	MITRA	MITRA$_\tau$
Adaptivity	++	-	-
Robustness and flexibility	++	++	++
Limited use of communication	++	-	++

TABLE 1.2: An overview of our contributions within the WSN domain in terms of the research requirements. The symbols have the following meaning: '+' ('++') means that the requirement is (strongly) satisfied, and '-' means the requirement is not satisfied, respectively.

4. **Long–term information collection in WSNs (Chapter 7):** Here, we propose a budget–limited MAB based energy management model for each agent within the network, in order to solve the energy management problem. For the routing problem, we propose two simple decentralised routing algorithms. The first is proveably optimal, but can sometimes use a large number of communication messages to coordinate the routing. The second algorithm is near–optimal, but its communication cost is significantly lower. By using one of the proposed routing algorithms, our approach can calculate the total amount of information throughput that the routing algorithm produces within that particular time step. This amount then forms the reward value that the MAB model receives by using the chosen energy budget allocation combination (see Section 7.4 for more details). With this reward value, the MAB model receives feedback about the efficiency of the chosen energy allocation combination, and thus, it can learn which combinations are more efficient ones. In more detail, these contributions are summarised in Table 1.2, and can be described as follows:

   - We devise the first multi–armed bandit learning based energy budget allocation approach, called MAB/EM. Based on this, we show how efficient energy management can be sustained in the long term, by using this approach.
   - We propose two simple decentralised routing algorithms, MITRA and MITRA$_\tau$. The former is the first to proveably maximise the total information throughput between layers of agents, whilst the latter has a near–optimal performance (it achieves, on average, 98% of the optimal solution), but with a reduced communication cost.
   - We empirically evaluate the performance of these algorithms through extensive simulations and show that information collection is increased by up to 120%, by applying the proposed algorithms, compared to that of USAC, a state–of–the–art method (see Section 7.1 for more details of USAC). Furthermore, we show that the communication cost of our approaches are low, compared to the cost of real data transmission.

These contributions have led to a number of peer-reviewed publications:

- L. Tran–Thanh, A. Chapman, J. E. Munoz De Cote Flores Luna, A. Rogers and N. R. Jennings (2010). Epsilon–First Policies for Budget–Limited Multi–Armed Bandits. In *Proceedings of the Twenty–Fourth AAAI Conference on Artificial Intelligence (AAAI–10)*, pp. 1211–1216, 2010.

- L. Tran–Thanh, A. Chapman, A. Rogers and N. R. Jennings (2012). Optimal Policies for Budget–Limited Multi–Armed Bandits. Accepted to *the Twenty–Sixth Conference on Artificial Intelligence (AAAI–12)*, 2012.

- L. Tran–Thanh, A. Rogers, and N. R. Jennings (2012). Long–Term Information Collection with Energy Harvesting Wireless Sensors: A Multi–Armed Bandit Based Approach. *Journal of Autonomous Agents and Multi-Agent Systems*, Volume 25, Issue 2, pp. 352–394.

The research results presented in the above publications are summarised and expanded upon by this thesis. To guide the reader through the remaining chapters, the following section contains a brief outline of the thesis structure.

1.4 Thesis Outline

The remainder of this thesis is structured as follows:

- Chapter 2 analyses the state–of–the–art in the multi–armed bandit literature. In particular, we describe the standard MAB model in more detail. Following this, we discuss the variants of MABs that focus on pulling costs and other pulling constraints. We then continue with the review of the unbounded knapsack literature, that forms the basis of our solutions in the subsequent chapters.

- Chapter 3 introduces our formal model of a budget–limited multi–armed bandit problem. Following this, we formulate our research objectives, with respect to the research requirements of this thesis.

- Chapter 4 discusses the budget–limited ε–first approach in more detail. In particular, we first introduce the pulling policy, then we discuss its computational complexity. Following this, we prove that its performance regret bound is typically a linear function of the budget. However, we also show that with a high probability, this regret bound can be improved to $O\left(B^{\frac{2}{3}}\right)$, where B is the budget size.

- Chapter 5 deals with the budget–limited upper confidence bound based approaches. Given this, we first introduce KUBE and fractional KUBE , and we discuss their

computational cost. We then provide upper bounds for their performance. We also show that their regret bounds are asymptotically optimal.

- Chapter 6 analyses the budget–limited decreasing ε–greedy based approaches. In more detail, we first discuss the pulling policies, namely KDE and fractional KUBE . We also discuss their computational cost. We continue the analysis by providing upper regret bounds for their performance, and showing that these bounds are also asymptotically optimal.

- Chapter 7 then contains the application of the budget–limited MAB model to the problem of long–term information collection problem within WSNs. We first present related work in this area, and detail why it does not meet all our requirements. Following this, we give the formal descriptions of our network model and research objectives. We then discuss our approach for efficient long–term data collection, which includes the budget–limited MAB learning based energy management method, and routing algorithms, respectively. Our approach is then empirically evaluated.

- Finally, Chapter 8 concludes and presents directions for future work to broaden the scope of our research and increase its practical applicability to the model of budget–limited bandits.

Chapter 2

Literature Review

In this chapter, we provide an overview of existing research studies against which our work is positioned. In order to do so, in the first part of the chapter (Section 2.1) we describe the standard, stochastic multi–armed bandit problem in more detail, and discuss the existing works on this bandit model. Following this, we focus on existing pulling policies of the standard MAB, that form the basis of our solutions in the subsequent chapters. We compare their performance from both a theoretical and an empirical perspective in Section 2.2, and we continue with the discussion of the variants of the standard bandit model in Section 2.3. In particular, we focus on bandit models that take several pulling constraints into account, and we further focus on these models that contain pulling costs or limited pulling abilities. Furthermore, as mentioned in Chapter 1, we use the unbounded knapsack approach in order to tackle our budget–limited MAB problem. Given this, we give a detailed overview of the knapsack literature in Section 2.5. More specifically, we first introduce the knapsack problem and its variants (including the unbounded knapsack) in Section 2.5.1. We then continue with solutions that efficiently tackle the unbounded knapsack (section 2.5.2). Finally, in Section 2.6, we conclude this chapter by summarising our findings and relating them back to our original research requirements (as detailed in Section 1.1).

2.1 The Stochastic Multi-Armed Bandit Problem

In this section, we describe the stochastic, or standard, MAB model (Robbins, 1952) in detail. In the MAB problem, there is a machine with K arms, each of which delivers rewards, that are independently drawn from an unknown distribution, when the machine's arm is pulled. A gambler must choose which of these arms to play. At each time step, he pulls one of the machine's arms and receives a reward (or payoff). The gambler's purpose is to maximise his return; that is, maximise the sum of the rewards he receives

over a sequence of pulls. As the reward distributions differ from arm to arm, the goal is to find the arm with the highest expected payoff as early as possible, and then to keep gambling using that best arm.

Now, to keep the terminology consistent with the multi-agent system based wireless sensor network problem considered in Chapter 7, hereafter we refer to the gambler as an agent. Thus, we can formulate the MAB problem as follows. Let K denote the number of the arms that the agent can pull. At each time step t, the agent pulls arm $i\,(t)$, which delivers the reward $r_{i(t)}\,(t)$, drawn from an unknown distribution of arm $i\,(t)$. Finally, let $T > 0$ denote the time horizon in which the agent operates. Thus, we have the following optimisation problem:

$$\max \sum_{t=1}^{T} r_{i(t)}\,(t). \tag{2.1}$$

Thus, the agent has to choose a policy, that is, a sequence of pulls, that may deliver the maximal reward at each time step t in order to achieve the maximum of equation 2.1. It is clear that if the distributions, from which the rewards are drawn, were known, the optimal policy would be to always pull the arm with the highest expected reward in order to maximise the cumulative rewards. Given this, in order to analyse the performance of a pulling policy we compare its performance with this theoretical optimal policy. In particular, we study the *regret* of the policy for not playing optimally. Now, let μ_i denote the *expected reward value* of arm i, where

$$\mu^* = \max_i \mu_i. \tag{2.2}$$

The regret $R^T\,(A)$ of pulling policy A after T pulls can be defined as:

$$R^T\,(A) = T\mu^* - \sum_{t=1}^{T} r_{i(t)}\,(t). \tag{2.3}$$

The MAB model, due to its clear representation of the trade–off between exploration and exploitation (see Chapter 1), has been used in a variety of areas. The historical motivation for this model was given by clinical trials where different treatments need to be experimented with, while patient loss should also be minimised as well (Hardwick and Stout, 1991). MAB models are also used to solve financial and investment problems as well. For example, optimal, but a priori unknown, investment options can be learned by using MAB exploration, while income maximisation is provided by MAB exploitation (Lai and Lim, 2005; Wang and Wang, 2009). In addition, MAB can also be adopted to the area of e–commerce, as an efficient way to identify the ranking of web documents (Radlinski *et al.*, 2008), or as a learning technique for optimising online advertisement (Chakrabarti *et al.*, 2008).

2.2 Stochastic Bandit Policies

Within this section, we discuss the bandit pulling policies in more detail. Recall that a fundamental dilemma in the MAB problem is the trade-off between exploration and exploitation. Specifically, if the agent exclusively chooses the action that it thinks is the best (i.e. exploitation), it may fail to discover that one of the other actions actually has a higher expected payoff. On the other hand, if he spends too much time trying out all the actions and gathering statistics (i.e. exploration), it may fail to choose the best action often enough to get a high return.

Against this background, researchers have proposed a variety of approaches that tackle this exploration-exploitation conflict from different aspects. Among these, the most simple policies are the greedy algorithm and its variants (Sutton and Barto, 1998; Vermorel and Mohri, 2005). In particular, the simplest policy in any bandit setting is a *greedy* policy (Sutton and Barto, 1998), which chooses the arm with the current highest estimated exptected reward at each time step. In most bandit problems, however, this algorithm demonstrates low efficiency, as the agent performs insufficient exploration (Sutton and Barto, 1998). Given this, a number of variants have been introduced in order to explicitly take exploration into account. One of the simplest approaches is the ε–greedy policy (Watkins, 1989), in which the agent follows the greedy policy (i.e. it pulls the arm with the highest estimate) with probability $(1 - \varepsilon)$, and it selects a random arm with probability ε. The value of ε is selected *a priori* and can be interpreted as an exploration parameter; that is, higher values correspond to more exploration and *vice versa*. The exploration parameter ε guarantees that every possible arm is continuously pulled as time goes by. This implies that ε–greedy fails to achieve asymptotic convergence to the optimal behaviour, since it is desirable to stop exploring once the optimal arm is learnt. Nevertheless, this policy typically performs well in finite time (i.e. when the running time horizon is finite), in a number of applications (Kuleshov and Precup, 2010; Sutton and Barto, 1998; Vermorel and Mohri, 2005).

Another variant of the greedy algorithm is the ε–first approach (Even-Dar *et al.*, 2002), which explicitly splits the exploration phase from exploitation. In particular, if the time horizon is T, then the agent randomly chooses an arm to pull (exploration) for the first εT time steps and then selects greedily for the remaining $(1 - \varepsilon)\, T$ steps. This policy ensures that all exploration is performed at the beginning when the agent has the highest levels of uncertainty regarding the expected rewards of each arm. Similarly to the ε–greedy, this policy does not converge to the optimal behaviour in general, since it might wrongly choose a suboptimal arm to pull within the exploitation phase. However, Even-Dar *et al.* showed that a high probability, the ε–first approach can achieve asymptotic convergence. This type of performance guarantee, which only holds with a certain probability, is referred to as the *probably approximately correct* (PAC) analysis (Valiant,

1984). Nevertheless, ε–first is found to outperform existing pulling policies in may applications (Kuleshov and Precup, 2010; Vermorel and Mohri, 2005). This is due to the fact that by focusing on pure exploration at the beginning, ε–first typically learns the optimal behaviour faster than other policies, that carry the exploration throughout the whole operating time.

To address the desire for asymptotic convergence, Auer *et al.* (2002) extended the previous two policies and proposed an algorithm, called *decreasing ε–greedy*, or ε_t–greedy, where the agent explores with probability $\min\{1, \varepsilon_t\}$ at time t and otherwise selects greedily. Here, $\varepsilon_t = \frac{C}{t}$ for some $C > 0$, and is decreasing as t grows. Beside the property of asymptotic convergence, Auer *et al.* also showed that decreasing ε–greedy has a strong finite–time performance. In particular, they proposed an $O(\ln T)$ upper bound for the performance regret of the policy, where T is the running time horizon. This performance bound is asymptotically optimal, as it only differs from the best optimal regret bound, that a pulling policy can achieve, by a constant factor. In fact, it can be shown that for any pulling policy, there exists a bandit setting, in which the performance regret of that particular policy is $\Omega(\ln T)$ (i.e. it is at least logarithmic) (Anantharam *et al.*, 1987; Lai and Robbins, 1985). In addition, decreasing ε–greedy shows good performance in experimental studies, compared to that of other policies (Kuleshov and Precup, 2010; Vermorel and Mohri, 2005). Specifically, it converges to the performance of ε–first and typically outperforms the others.

Apart from the variants of the greedy approach, other pulling techniques focus on theoretical guarantees. In more detail, Lai and Robbins (1985) proposed a policy, which they called *uniformly good policy*. This achieves logarithmic regret bounds for some specific families of probability distributions (including exponential families), as the time horizon tends to infinity. The regret bound's constant factor is based on the Kullback–Leibler divergence (Lai and Robbins, 1985), and this bound guarantees that the algorithm satisfies the property of asymptotic convergence. Their result was later improved by Anantharam *et al.* (1987) and Agrawal (1995b). In particular, Anantharam *et al.* extended it to bandit models where multiple arms can be pulled at the same time. Agrawal later proposed a class of algorithms that is probability distribution independent; that is, it does not contain any restriction on the distributions of the rewards. To do so, they applied a concept called *optimism in the face of uncertainty* (OFU), first introduced by Kaelbling (1993), that allows the agent to select the arms by using a combination of the estimates of the expected reward values and the uncertainty of those reward estimates, such that arms with high uncertainty are selected more often. By so doing, Agrawal proved that the proposed algorithms are thus much easier to compute than Lai and Robbins'. In addition, they can still achieve the same asymptotic optimal regret bound, but with a significantly larger constant factor.

To provide finite–time regret bounds, Auer *et al.* (2002) enhanced Agrawal's technique by designing a class of policies called *upper confidence bound* (UCB). These approaches achieve bounded regret in finite time as well as having optimal asymptotic convergence. In particular, they first proposed *UCB*1, which can be described as follows. UCB1 pulls each arm once at the beginning, then at each subsequent time step t, UCB1 selects arm i that maximises:

$$\hat{\mu}_{i,t} + \sqrt{\frac{2 \ln t}{n_{i,t}}}, \tag{2.4}$$

where $\hat{\mu}_{i,t}$ is the estimate of arm i's expected reward value, and $n_{i,t}$ is the number of times UCB1 pulled arm i until time step t. This policy achieves $O(\ln T)$ finite–time regret bound, but the constant factor of the bound is significantly larger than that of Lai and Robbins' method. In addition, it was found to perform poorly in finite-time applications (Auer *et al.*, 2002). To improve the performance of UCB1 in real–world applications, Auer *et al.* modified the policy to *UCB–tuned* so that the latter pulls the arm that maximises:

$$\hat{\mu}_{i,t} + \sqrt{\frac{2 \ln t}{n_{i,t}} \min\left\{\frac{1}{4}, V_i(n_{i,t})\right\}}, \tag{2.5}$$

where

$$V_i(n_{i,t}) = \frac{1}{n_{i,t}} \sum_{\tau=1}^{n_{i,t}} r_i^2(t_\tau) - \hat{\mu}_{i,t} + \sqrt{\frac{2 \ln t}{n_{i,t}}}. \tag{2.6}$$

Here, $r_i(t_\tau)$ denotes the reward value we get by pulling arm i at time step t_τ (i.e. the time step in which we pull arm i the τ^{th} time). Although UCB–tuned shows good performance in applications, it does not have any theoretical guarantees such as finite–time regret bound or asymptotic convergence (Auer *et al.*, 2002).

In addition, Auer *et al.* (2002) also proposed UCB2, a variant of UCB1, with the purpose of decreasing the large constant factor within the regret bound. This UCB–based policy can be described as follows. The policy chooses an arm, and pulls it for an epoch (i.e. a specified interval of time). In so doing, it maintains an index τ_i for each arm i, that denotes the starting time of the τ^{th} epoch in which we pull arm i. Similar to UCB1, it pulls each arm once at the beginning, setting $\tau_i = 0$ for all i. Following this, at each epoch, the agent chooses an arm that maximises:

$$\hat{\mu}_{i,t} + \sqrt{\frac{(1+\alpha) \ln\left(\frac{et}{l(\tau_i)}\right)}{2 l(\tau_i)}}, \tag{2.7}$$

where

$$l(\tau_i) = \lceil (1+\alpha)^{\tau_i} \rceil. \tag{2.8}$$

Here, the length of epoch τ_i (i.e. the τ^{th} epoch in which we choose arm i to pull) is $l(\tau_i) - l(\tau_i - 1)$. In addition, e is Euler's number and α is a tuning parameter, that has

to be set *a priori*. Finally, at the end of epoch τ_i, we increase the value of τ_i by 1.

Note that the UCB approach is computationally less expensive than Lai and Robbins' algorithm. However, it has a lower efficiency in terms of regret bounds. In particular, the asymptotic constant factor within the regret bound of Lai and Robbins' algorithm is tighter than that of the UCB. Given this, a range of more recent research work focuses on improving the constant factor within the logarithmic regret bound of the UCB. In particular, Auer and Ortner (2010) revisited the UCB approach and showed that UCB has inefficient regret bounds if the real expected reward values of the arms are close to each other (i.e. it is hard to learn the optimal arm). They proposed an extension of UCB that shows improvement in terms of providing tighter regret bounds than that of the UCB. More recently, Maillard *et al.* (2011) derived a logarithmic regret bound that contains a Kullback–Leibler divergence based constant factor, which is proven to be near–optimal (i.e. it is as tight as Lai and Robbins' regret bound). In so doing, the authors improved the UCB based technique described in the work of Honda and Takemura (2010). However, by improving the regret bound, the algorithm becomes significantly more costly in terms of computational complexity.

Similar to the UCB policies, the POKER (for price of knowledge and estimated reward) algorithm also follows the concepts of interval estimation and OFU. In particular, let $\mu^* = \max_i \mu_i$ denote the optimal expected reward value. Now if $I(t)$ denotes the arm with the highest estimated reward mean at time step t such that:

$$I(t) = \arg\max_i \hat{\mu}_{i,t}, \tag{2.9}$$

then we have $\hat{\mu}_{I(t)} = \max_i \hat{\mu}_{i,t}$. Now, let δ_t denote the expected reward improvement at time step t, which can be defined as $\delta_t = \mathbf{E}\left[\mu^* - \hat{\mu}_{I(t)}\right]$. At each time step t, the agent chooses an arm i that maximises:

$$\hat{\mu}_{i,t} + \delta_t P\left[\mu_i - \hat{\mu}_{I(t)} \geq \delta_t\right](T-t), \tag{2.10}$$

where $P\left[\mu_i - \hat{\mu}_{I(t)} \geq \delta_t\right]$ is the probability that by choosing arm i, the reward improvement will be higher than δ_t. Intuitively, the product in the second term of Equation 2.10 can be viewed as an estimate of the knowledge acquired if arm i is pulled repeatedly until T. However, both δ_t and $P\left[\mu_i - \hat{\mu}_{I(t)} \geq \delta_t\right]$ are not known *a priori*. Within POKER, the agent uses heuristics in order to estimate these values. In so doing, it first takes the decreasing order of the arm's estimated reward mean values such that $\hat{\mu}_{i_1,t} \geq \hat{\mu}_{i_2,t} \geq \ldots \geq \hat{\mu}_{i_q,t}$, where q is the number of arms that have been pulled at least once until time step t. Now, the agent estimates δ_t by using the following approximation technique:

$$\delta_t = \frac{\hat{\mu}_{i_1,t} - \hat{\mu}_{i_{\sqrt{q}},t}}{\sqrt{q}}. \tag{2.11}$$

To estimate the reward improvement probability, let $\hat{\sigma}_{i,t}$ denote the estimated standard deviation of arm i's reward distribution. Thus, noting that

$$P\left[\mu_i - \hat{\mu}_{I(t)} \geq \delta_t\right] = P\left[\mu_i \geq \hat{\mu}_{I(t)} + \delta_t\right], \tag{2.12}$$

which is estimated by

$$\int_{\hat{\mu}_{I(t)}+\delta_t}^{\infty} \mathcal{N}\left(x, \hat{\mu}_{i,t}, \frac{\hat{\sigma}_{i,t}}{\sqrt{n_{i,t}}}\right) \mathrm{d}x, \tag{2.13}$$

where $\mathcal{N}\left(x, \hat{\mu}_{i,t}, \frac{\hat{\sigma}_{i,t}}{\sqrt{n_{i,t}}}\right)$ denotes a normal distribution with expected value $\hat{\mu}_{i,t}$ and standard deviation $\frac{\hat{\sigma}_{i,t}}{\sqrt{n_{i,t}}}$. Here, $n_{i,t}$ denotes the number of times the agent has pulled arm i until time step t. By using the abovementioned heuristics, Vermorel and Mohri (2005) proved that POKER asymptotically converges to the optimal policy as time goes by. On the other hand, they could not provide a finite–time regret bound. More recently, Sykulski (2011) demonstrated that POKER shows poor performance within experimental studies, compared to that of simpler policies such as ε–greedy or ε–first.

Apart from the abovementioned approaches, an alternative way to find a trade–off between exploration and exploitation is to randomly pull the arms such that the arms that are expected to have higher rewards are selected with higher probability. This concept is usually denoted as the *probability matching* technique (Vermorel and Mohri, 2005). In particular, the first of this kind is *SoftMax*, proposed by Luce (1959), where at each time step t, arm i is chosen with probability

$$p_i\left(t\right) = \frac{e^{\frac{\hat{\mu}_{i,t}}{\tau}}}{\sum_{j=1}^{K} e^{\frac{\hat{\mu}_{j,t}}{\tau}}}, \tag{2.14}$$

where τ is a tuning parameter, which determines the degree of exploration. In particular, large values of τ correspond to more equal weighting between the arms, and thus, more exploration. On the other hand, as $\tau \to 0$, SoftMax converges to the greedy algorithm (i.e. pure exploitation). By suitably choosing the value of τ, SoftMax can ensure that arms that are likely to be suboptimal are less frequently selected, compared to arms with high reward values. However, since τ does not change over time, the overall degree of exploration does not change either. This leads to poor theoretical performance of SoftMax; that is, there is no guarantee that SoftMax will satisfy the property of asymptotic convergence (similarly to the case of ε–greedy). To address this drawback, Cesa-Bianchi and Fischer (1998) proposed the *SoftMix* algorithm in which the value of τ is decreasing over time, in a similar vein to decreasing ε–greedy (it is typically decreased at rate $\frac{1}{t}$, or $\frac{\ln(t)}{t}$). They also provided a $O\left(\ln^2\left(T\right)\right)$ finite–time regret bound, which implies the asymptotic converging behaviour of SoftMix (since the regret is sub–linear). However, this regret bound is less efficient, compared to the optimal logarithmic bounds. Another popular method is the *exponential weight algorithm for exploration*

	Computational Cost	Experimental Performance	Asymptotic Convergence	Finite–Time Bound	Constant Factor
ε–first	++(*)	++(*)	-	-	-
ε–greedy	++	++	-	-	-
decreasing ε–greedy	++	++	Yes	$O(\ln T)$	large
Lai and Robbins'	+	-	Yes	-	small(*)
Agrawal's	++	-	Yes	-	large
UCB1	++	+	Yes	$O(\ln T)$	large
UCB–tuned	+	++	-	-	-
UCB2	++	+	Yes	$O(\ln T)$	moderate
Improved UCB	+	-	Yes	$O(\ln T)$	small
Maillard *et al.*'s	+	-	Yes	$O(\ln T)$	small(*)
POKER	+	+	Yes	-	-
SoftMax	++	++	-	-	-
SoftMix	++	++	Yes	$O\left(\ln^2 T\right)$	large
Exp3	+	+	Yes	$O\left(\sqrt{T}\right)$	large

TABLE 2.1: An overview of the pulling policies in the bandit domain. The symbols have the following meaning: '+' ('++') means that the property is (strongly) satisfied. In addition, '(*)' indicates the best performance within a row. On the other hand, '-' means the property is not known.

and exploitation (Exp3), proposed by Auer *et al.* (2003). In particular, at each time step t, $p_i(t)$ is calculated as follows:

$$p_i(t) = (1-\gamma)\frac{w_i(t)}{\sum_{j=1}^{K} w_j(t)} + \frac{\gamma}{K}. \tag{2.15}$$

Here, $\gamma \in (0,1]$ is a tuning parameter, and $w_i(t)$ are the probability weights, that can recursively be calculated as follows. For each $i \in \{1,2,\ldots,K\}$, $w_i(1) = 1$, and if arm i is pulled at time step t, we have

$$w_i(t+1) = w_i(t)\exp\left(\gamma\frac{r_i(t)}{p_i(t)K}\right). \tag{2.16}$$

Otherwise, we have $w_i(t+1) = w_i(t)$. Note that the tuning parameter γ determines the degree of exploration. In particular, $\gamma = 1$ yields pure random exploration, and $\gamma \to 0$ brings Exp3 towards the pure exploitation approach. This algorithm, however, is designed for tackling non–stochastic bandit problems (see Section 2.3 for more details), and thus, shows inefficient performance in stochastic bandit settings. More specifically, it can achieve $O\left(\sqrt{T}\right)$ regret bound, which is significantly less than the optimal logarithmic regret bound (Auer *et al.*, 2003).

In summary, the comparison of the abovementioned policies is depicted in Table 2.1. In particular, we can see that at one extreme are the simple and experimentally efficient

policies such as ε–first and ε–greedy. However, these algorithms cannot guarantee theoretical efficiency (i.e. asymptotic convergence and finite–time regret bounds). On the other hand, UCB–based algorithms (e.g. UCB1, UCB2, or Maillard *et al.*'s algorithm) achieve efficient theoretical performance, but with poor experimental results. Other algorithms typically show significant shortfalls, compared to the abovementioned algorithms. In particular, they either have poor performance in practice, compared to that of the ϵ–first, or they provide significantly worse theoretical guarantees, compared to that of the UCB–based approaches. A notable exception, however, is the decreasing ε–greedy policy. More precisely, this algorithm approaches the performance of the ϵ–first in practice, and shows similar theoretical results, compared to that of the UCB algorithms. Given this, it acts as a trade–off between the simple, but experimentally efficient, approaches and the theoretically more advanced, but experimentally poorly performing, algorithms. Since these algorithms do not take pulling cost into account, they are not suitable to the budget–limited multi–armed bandit problem. However, they still form the foundations on which we can rely, in order to efficiently tackle the budget–limited MAB. In particular, within Chapters 4, 5, and 6, we will introduce three types of budget–limited MAB algorithms, based on the ε–first, UCB, and the decreasing ε–greedy approaches of the standard MAB.

2.3 Bandit Variants

Given the detailed description of the stochastic bandit model and its pulling policies in the previous section, we now consider a number of variants of the MAB model. Although these variants typically do not show similarities to the budget–limited MAB, many of them may form the basis of our future work. The bandit model can be varied from a number of aspects, such as varying the set of arms, the behaviour of the rewards, or additional information that the agent can take into account. These variants are covered in detail within this section as follows. In Section 2.3.1, we first describe the bandit variants that consider different types of available arms to pull. We then continue with bandit models that vary the nature of the reward values in Section 2.3.2. In addition, Section 2.3.3 focuses on bandit models where additional information is also available (beside the reward value that the agent receives by pulling a particular arm).

2.3.1 Set of Arms

One way to extend the multi–armed bandits is to allow *infinitely* many arms (i.e. *continuum–armed bandits*), instead of limiting the arms to a finite set (Agrawal, 1995a; Auer *et al.*, 2007; Bubeck *et al.*, 2011; Cope, 2009; Kleinberg, 2005). Such problems can be found, for example, in control theory, where the agent has to find an optimal

parameter setting from a continuous parameter space. Other examples include, but are not limited to product pricing, transmission power controlling, and temperature optimisation in chemical processes (Bubeck *et al.*, 2011; Cope, 2009). This problem was first discussed by Agrawal (1995a), and a pulling policy with near optimal regret bounds was provided by Kleinberg (2005). This result was later improved by Auer *et al.* (2007). More recently, Kleinberg *et al.* (2008) generalised the continuum–armed bandit problem to the bandit model in metric spaces. Based on this result, Bubeck *et al.* (2011) extended the model so that the set of arms is allowed to be a generic measurable space and the mean–payoff function is locally Lipschitz continuous [1]. Although we focus on the case of finite set of arms in this thesis, we consider the continuum-armed variant as a possible way to extend our work, and thus, the aforementioned works may found the basis of future investigations within the budget–limited MAB domain.

Another way to vary the set of arms is to allow the agent to pull more than one arm at the same time (Anantharam *et al.*, 1987; Cesa-Bianchi and Lugosi, 2009). More specifically, Anantharam *et al.* (1987) allows the agent to pull $m \geq 1$ arms at the same time ($m < K$). Based on the technique introduced by Lai and Robbins (1985), Anantharam *et al.* proposed a pulling policy that proveably achieves logarithmic regret bounds (i.e. the bound is asymptotically optimal). More recently, Cesa-Bianchi and Lugosi (2009) introduced the *combinatorial bandit* problem in which the agent chooses a finite combination of arms to pull. However, the number of arms that the agent can pull at the same time, is limited by a threshold $L < K$. Cesa-Bianchi and Lugosi provided an algorithm, called *ComBand*, that achieves $O\left(L\sqrt{TK\ln N}\right)$ regret bound, where N is the number of available combinations of arms that the agent can pull at each time step. Within our settings, we only allow the agent to pull one single arm at each time step. Given this, this variant of MAB is out of our scope.

Apart from the abovementioned variants, other research work allows the set of arms to change over time (Chakrabarti *et al.*, 2008; Wittle, 1981). In more detail, Wittle (1981) studied the *arm acquiring bandits*, in which new arms are available to the agent during the operating time. On the other hand, Chakrabarti *et al.* (2008) discussed the *mortal bandit* problem that allows the arms to be deleted. These models are motivated by a number of applications in which new options can arrive or disappear from the system (e.g. online advertisement, or financial investments). Note that within the budget–limited models which we focus on in this thesis, the set of feasible arms (i.e. arms that we can pull with respect to the remaining budget) decreases over time as the pulling budget decreases. However, within mortal bandits, it is known *a priori* which arm becomes infeasible in the future, while within the budget–limited model, this is not known in advance, since it depends on the sequence of pulls in the past. Hence, an arm might become infeasible to be pulled at time t if we choose a particular pulling sequence,

[1] A function $f(x)$ is locally Lipschitz continuous if for every x from its domain, there is a neighbourhood $U(x)$ of x and a constant k such that if $x_1, x_2 \in U(x)$, then $|f(x_1) - f(x_2)| \leq k\,|x_1 - x_2|$.

and the same arm might be still feasible, if we choose another sequence of pulls. This indicates that both arm acquiring and mortal bandit models are not suitable to describe our budget–limited bandit problem.

2.3.2 Nature of Rewards

In the stochastic bandit model, the rewards are randomly chosen from unknown, but fixed distributions. However, many real–world applications require non–stationary behaviour; that is, the reward distributions may vary over time. This situation typically occurs in systems where the environment is dynamic (i.e. the environmental characteristics vary over time), such as wireless sensor networks, financial markets, or dynamic controlling systems. In this spirit, Wittle (1988) introduced the *restless bandit* problem, in which the *state* of each unselected arm changes over time. In more detail, the reward distribution of the arms dynamically change as time goes by, if they are not selected for pulling. More recently, a number of researchers focus on bandit within *piece–wise stationary environments* (DaCosta *et al.*, 2008; Hartland *et al.*, 2006). In particular, these models assume that the reward distributions are piece–wise stationary; that is, they are stationary within certain time intervals. This assumption is driven by the fact that many real–world applications follow this behaviour (i.e. temperature change, or behaviourial changes in habits of animals). Given this, Hartland *et al.* (2006) proposed *Adapt–EvE* (for adaptive exploration and exploitation), a pulling policy that adapts to the environmental changes. In particular, it uses a two–level bandit model, in which the lower level is a standard stochastic MAB, while the meta–bandit level is dedicated to detecting the changes within the environment. If the meta–bandit finds signs of change (e.g. by using statistical tools), it resets the standard MAB. Similarly, D-MAB (for dynamic multi–armed bandit), proposed by DaCosta *et al.* (2008), also uses change detection to reset the underlying bandit algorithm. Since we do not take dynamic bandits into consideration, these works are out of our focus.

Another way to modify the reward generating mechanism is to assume that it is *non–stochastic*, or *adversarial* (Auer *et al.*, 2003). In more detail, within this bandit setting, the agent plays a finite repeated game against nature, or an *adversary*, in which at each time step t, the adversary generates a vector of rewards $\mathbf{v}(t) = \{v_i(t)\}_{1 \leq i \leq K} \in \mathbb{R}^K$, which is unknown to the agent. Note that K is the number of arms that the agent can pull. Following this, the agent chooses an arm $i \in \{1, \ldots, K\}$ to pull, and receives reward $v_i(t)$. The goal here is also to maximise the total reward that the agent can achieve over time horizon T. We say that the adversary is *oblivious* if the choice of the reward vector $\mathbf{v}(t)$ is independent from the previous pulls of the agent, and it is *non–oblivious* if the adversary takes the pulling history into account. As the latter case forms an extremely hard problem to tackle (Bubeck, 2010), most research work focuses on the former. Specifically, Auer *et al.* (2003) proposed Exp3 to tackle this bandit problem (see

Section 2.2 for more details). In particular, they provided a $O\left(\sqrt{T}\right)$ regret bound for Exp3. Note that here, analogously to the stochastic bandit, the regret is measured as the difference between the performance of the algorithm and that of a best single arm policy (i.e. a policy that repeatedly pulls a single arm). The results of Auer *et al.* were later improved by Bubeck (2010). Although this bandit variant appears flexible, it is not always practical, since in many applications, the environment does not behave in an adversarial way. Within this thesis, we do not focus on the adversarial aspect of the budget–limited MAB. However, it can be regarded as future work (see Chapter 8 for more details).

2.3.3 Additional Information

Within the standard bandit problem, the sole feedback that the agent receives from the system is the reward value of the chosen arm. However, in a number of real–world applications, agents are likely to have additional side information (e.g. information from what they have observed, or information given by other participants in the system) that is received throughout their operating time. This side information can be regarded as additional information (other than observed rewards) that is related to, but does not fully reveal, the expected rewards of future pulls (Sykulski, 2011). This bandit variant is usually referred to as *bandits with covariates* (Clayton, 1989; Pavlidis *et al.*, 2008; Rigollet and Zeevi, 2010; Woodfoofe, 1982) or *contextual bandits* (Beygelzimer *et al.*, 2011; Langford and Zhang, 2007; Lu *et al.*, 2010). Within this bandit setting, at each time step t, the agent observes a noisy context (or covariate) $X(t)$ (i.e. side information), that is randomly drawn from a known and fixed probability distribution P_X. Let $r_i(t)$ denote the reward the agent receives if arm i is pulled at t. We have:

$$\mathbf{E}\left[r_i(t)\,|X(t)\right] = f^{(i)}\left(X(t)\right), \tag{2.17}$$

where $f^{(i)}$ are not revealed to the agents *a priori*. Within the contextual bandit setting, Woodfoofe (1982) studied this problem in the one–armed bandit version, while Rigollet and Zeevi (2010) covered the two–armed bandit model. The multi–armed model was discussed in Beygelzimer *et al.* (2011); Langford and Zhang (2007) and Lu *et al.* (2010). In particular, Langford and Zhang (2007) proposed the *epoch–greedy* algorithm that achieves $O\left(T^{\frac{2}{3}}\right)$ regret bound. Beygelzimer *et al.* (2011) improved this result by proposing *Exp4.P*, a pulling policy that achieves $O\left(\sqrt{KT}\right)$ regret bound with high probability. In addition, Lu *et al.* (2010) extended the model to more general metric spaces. Within this thesis, we do not focus on the additional information that the agent can receive, and thus, we assume that there is no side information within the budget–limited MAB. Given this, the abovementioned approaches are not suitable for

the budget–limited bandit model, since they are mainly designed to tackle the challenges of having side information. However, the budget–limited MAB can be extended by adding side information into the model. Thus, one of the possible future work is to combine our results with the aforementioned methods from the domain of contextual bandits.

Apart from the contextual bandits, another way to take additional information into account is to reveal the reward value of arms that are not pulled as well (Cesa-Bianchi *et al.*, 1997). This can be seen as a finite repeated game with *full information*. Within this setting, the best possible regret bound is $\sqrt{T \ln K}$ (for more details see Chapter 3 in Bubeck (2010)). A variant of this full information problem is the *label efficient prediction* problem (Allenberg *et al.*, 2006; Cesa-Bianchi *et al.*, 2005; Ottucsák, 2007). Here, at each time step the agent can ask to see the rewards of other arms that are not pulled. However, the agent can ask for side information for at most than m times over its operating time. Within this version, Cesa-Bianchi *et al.* (2005) proposed a $O\left(T\sqrt{\frac{K}{m}}\right)$ regret bound, which was later improved by Bubeck (2010) to $O\left(T\sqrt{\frac{\ln K}{m}}\right)$. Similarly to the contextual bandits, bandits with full information can also be extended by adding pulling costs into the model. However, the analysis of this combination of models remains as future work.

2.4 Bandits with Pulling Cost

A common theme in the abovementioned bandit variants is that pulling the arms is not costly, and thus, any arm can be pulled arbitrary many times during the agent's operating time. However, in many real–world applications, making a decision (i.e. choosing an arm to pull) is costly (see Chapter 1 for more details). By ignoring pulling costs, the agents can explore without limits. In contrast, when pulling costs are taken into consideration, exploration has to be done more carefully, otherwise it results in inefficient performance (Farias and Madan, 2011; Madani *et al.*, 2004). Against this background, several recent research works have focused on bandit versions with some costs that are related to arm pulling (Agrawal *et al.*, 1988; Bubeck *et al.*, 2009; Farias and Madan, 2011; Guha and Munagala, 2009; Madani *et al.*, 2004). In particular, Agrawal *et al.* (1988) considered the bandit problem with *switching costs*, where the switching of arms between subsequent pulls is costly, and the agent has to pay a fixed cost value C. Here, the total reward regret is combined with the total switching cost in order to form the cumulative regret. The objective of this bandit problem is then to minimise this cumulative regret. In so doing, Agrawal *et al.* proved that the best possible regret bound is logarithmic, and they also provided an efficient pulling policy that asymptotically achieves this bound. More recently, Guha and Munagala (2009) extended this bandit model so that the switching cost is measured by a metric. In more detail, if $i(t) = i$ (i.e.

the arm pulled at time step t) and $i(t+1) = j$ then the switching cost is a fixed value $l_{i,j}$. Within this variant, Guha and Munagala proposed an efficient pulling algorithm that approximates the optimal solution by a $(3+\varepsilon)$ constant factor, where $\varepsilon > 0$ is an arbitrarily small number.

Similar to bandits with switching cost, the *irrevocable* bandit model also aims to minimise the cost of switching arms between subsequent pulls (Farias and Madan, 2011). However, while bandits with switching cost allow re–switching (i.e. return to pull an arm that has been pulled a long time ago), in the irrevocable bandit model, the agent is not allowed to return to an arm once it switched from that arm. This restriction is inspired by a number of financial and economical applications (e.g. retail selling, or fashion designing), in which switching business partners causes significant loss in trust, and thus, the option of returning to that partner is not possible (Farias and Madan, 2011).

The bandit with switching cost and irrevocable arms, however, does not share the same issues of the budget–limited MAB, since there is no budget limit for the total pulling cost. In particular, the algorithms designed for these bandit models may fail to achieve good performance within the budget–limited MAB, since they might result in sequence of pulls in which the total pulling cost exceeds the budget limit.

Another way to take pulling cost into account is to set a budget limit for the total pulling cost. In particular, Madani *et al.* (2004) introduced a *budgeted* bandit version in which arm pulling is costly, and different arms have different costs. In addition, the total pulling cost cannot exceed a given budget B. This model shows similarities to our model, however, it solely focuses on exploration, ignoring the exploitation phase. In particular, the budgeted bandit problem consists of a budget limited exploration phase, and an unlimited exploitation phase. Within the exploration phase, pulling arms is costly, and the total cost of exploration cannot exceed the budget. This type of problem can be found in many applications, such as clinical trials (i.e. we have to determine the best medicine, but trials are costly), or transmission power optimisation between wireless devices (i.e. we have to find the optimal value of transmission power, but each trial consumes energy). Within their work, Madani *et al.* showed that this problem is NP–hard, and they proposed an approximation algorithm with poor efficiency. This result was then later improved by Guha and Munagala (2007), who proposed an approximation policy with approximation factor 4. In their subsequent work, Guha and Munagala (2009) further improved this approximation factor to $(3+\varepsilon)$.

More recently, Bubeck *et al.* (2009) addressed a variant of the budgeted bandit problem, in which the pulling cost is considered to be the same for every arm, but the budget is not known *a priori* to the agent. Given this, the authors focus on *anytime* policies that have an incremental improvement in terms of finding the best arm as time goes by.

Note that the anytime property is crucial here, since the budget, and thus, the stopping time, is unknown. In so doing, Bubeck *et al.* introduce a new regret, called *simple regret*, which can be described as follows. After the exploration budget is exceeded the agent chooses a single arm that it assumes to be the best arm to repeatedly pull in the exploitation phase. Let $J(t)$ denote the arm that the agent chooses after t time steps. The simple regret is defined as:

$$r_t = \mu^* - \mu_{J(t)}, \tag{2.18}$$

where μ^* denotes the highest expected reward. Against this background, Bubeck *et al.* (2009) proposed a number of exploration algorithms and recommendation strategies (i.e. a strategy to choose the best arm). In particular, they provided a $O\left(\sqrt{\frac{K \ln K}{t}}\right)$ simple regret bound, where t is the number of time steps after which the exploration phase has to stop.

Finally, Antos *et al.* (2008) considered a similar model, in which they aim to estimate the expected reward of all arms, in order to determine the best arm with high certainty. In so doing, they introduced the concept of *active learning regret*, which is formalised as:

$$\max_{i \in \{1,\dots,K\}} \mathbf{E}\left[(\mu_i - \hat{\mu}_{i,t})^2\right], \tag{2.19}$$

where $\hat{\mu}_{i,t}$ denotes the estimate of the expected reward of arm i at time step t. In particular, Antos *et al.* proposed an algorithm that achieves $O\left(T^{-\frac{3}{2}}\right)$ regret bound in terms of active learning regret.

These models, although they show similarities to the budget–limited MAB, have different objectives; that is, to minimise the simple regret, and the active learning regret, respectively. This indicates that the provided pulling policies are not suitable for our model. More specifically, Bubeck *et al.* (2009) showed that a pulling policy that provides efficient performance in minimising the simple regret is not suitable in minimising the total regret. However, note that these models focus on the exploration phase only, and thus, they can be regarded as a part of our ε–first approach (Chapter 4), since this approach explicitly separates exploration from exploitation. Given this, we can use the aforementioned techniques to explore within the exploration phase of the ε–first approach. In particular, we indeed use the uniform pull policy, which is also recommended by Bubeck *et al.* (2009), to explore in the ε–first approach (see Chapter 4 for more details).

2.5 The Unbounded Knapsack Problem

Within the previous sections, we described the stochastic multi–armed bandit model and its variants, and discussed their connections with the budget–limited bandit problem.

We now turn to the description of the knapsack problems that form the basis of our approach in the subsequent chapters. In particular, we first describe a number of knapsack models, including the unbounded knapsack, in more detail (Section 2.5.1). We then discuss the solutions of the unbounded knapsack problem in particular in Section 2.5.2.

2.5.1 Knapsack Models

The standard knapsack problem and its variants are among the most well–known and widely applied optimisation problems (Marcello and Toth, 1990; Skiena, 1999), and can be defined as follows. A knapsack of weight capacity C is to be filled with some combination of K different items. Each item $i \in K$ has a corresponding value v_i and weight w_i, and the problem is to select a *subset* of items that maximises the total value of items in the knapsack, such that their total weight does not exceed the knapsack capacity C. Formally, we have to solve the following optimisation problem:

$$\text{maximise} \quad \sum_{i=1}^{K} x_i v_i, \tag{2.20}$$

$$\text{subject to} \quad \sum_{i=1}^{K} x_i w_i \leq C, \tag{2.21}$$

$$\forall i \in \{1, \ldots, K\} : \; x_i \in \{0, 1\}. \tag{2.22}$$

Note that each variable x_i is binary, since we can either choose the item i or not. The knapsack problem can be found in a large variety of research areas, such as task allocation (e.g. computer job scheduling), logistics (e.g. airline cargo dispatching), and financial investments (e.g. portfolio optimisation) (Kellerer *et al.*, 2004; Marcello and Toth, 1990).

In order to fit real–world applications, a variety of extensions and modifications have been made to the standard knapsack model. In particular, these modifications include, but are not limited to, extending (i) the domain of items (i.e. modifying Equation 2.22); (ii) the knapsack capacity (i.e. modifying Equation 2.21), or (iii) the objective (modifying Equation 2.20). Among these, one of the the most well–known knapsack variant is the *unbounded knapsack problem*, where more than one *identical copy* from the different *item types* are allowed. That is, x_i can be an arbitrary (non–negative) integer. The

unbounded knapsack can be formalised as follows:

$$\text{maximise} \quad \sum_{i=1}^{K} x_i v_i, \tag{2.23}$$

$$\text{subject to} \quad \sum_{i=1}^{K} x_i w_i \leq C, \tag{2.24}$$

$$\forall i \in \{1, \ldots, K\} : \; x_i \geq 0, \; x_i \text{ integer}. \tag{2.25}$$

Here, v_i and w_i are the value and the weight of item type i. That is, items from the same type have the same weight and value.

Apart from the unbounded knapsack, there are a variety of other knapsack models. This includes, but is not limited to, the following: bounded knapsack, d–dimensional knapsack, multiple knapsack, and quadratic knapsack (Kellerer *et al.*, 2004; Marcello and Toth, 1990). However, these models are out of scope of this thesis, and thus, we ignore their description (for more details of these models, see Kellerer *et al.* (2004)).

2.5.2 Algorithms for the Unbounded Knapsack

Given the knapsack models described in the previous section, we now focus on the unbounded knapsack in more detail. Thus, within this section, we discuss the algorithms for the unbounded knapsack problem, as they form the basis of our approaches to tackle the budget–limited MAB. As the unbounded knapsack problem is NP–hard (Andonov *et al.*, 2000; Kellerer *et al.*, 2004), the algorithms can be categorised to *exact* or *approximation* approaches. The former are optimal, but with increased computational costs, while the latter provide near–optimal solutions with low computational complexity.

In more detail, the exact algorithms typically use the technique of dynamic programming (Bellman, 1957) to exploit the observation that if a solution of the unbounded knapsack is optimal, then by removing an item r from the optimal knapsack packing, the remaining solution has to be optimal for the modified knapsack problem with capacity $C - w_r$, where C is the capacity of the original problem. Given this, the dynamic programming technique first solves the problem for a subset of item types. Then it adds a new item type to the subset, and checks whether the optimal solution has to be modified for the enlarged subset. This approach, however, is typically pseudo–polynomial in terms of computational cost, and its running time significantly depends on the sequence of item types added to the subset (Kellerer *et al.*, 2004; Marcello and Toth, 1990). Against this background, a number of researchers addressed this issue by providing efficient ways of choosing the next item to add to the subset (Andonov *et al.*, 2000; Babayev and Mardanov, 1994; Dudzinski, 1991; Marcello and Toth, 1990). The common approach is to use a dominance relationship between the items. For example, a frequently used

Algorithm 2.1 Fractional Unbounded Knapsack based Algorithm

1: $I^* = \arg\max_{i \in \{1,\dots,K\}} \{\frac{v_i}{w_i}\}$;
2: **while** packing is feasible **do**
3: pack item from type I^*;
4: **end while**

dominance relationship is the *collective dominance*, which can be defined as follows: A *set of item types I collectively dominates item type $i \notin I$* if there exists a vector $\mathbf{y} = \langle y_1, \dots, y_K \rangle$ such that

$$\sum_{j \in I} y_j w_j \leq w_i \quad \text{and} \quad \sum_{j \in I} y_j v_j \geq v_i. \tag{2.26}$$

That is, by substituting an item of type i with the combination $\mathbf{y}$ of set of types I, we can increase the total value, while the total weight is not increased (i.e. set I collectively dominates item type i). Note that there are other types of dominance that are used in state–of–the–art dynamic programming based algorithms, such as simple, multiple, or threshold dominance (for more details see Andonov *et al.* (2000); Kellerer *et al.* (2004)). Now, in order to decrease the computational complexity of the dynamic programming approach, the next item type to be added to the subset is always the item type that is not dominated by any other types that are still not in the subset (Andonov *et al.*, 2000; Babayev and Mardanov, 1994; Kellerer *et al.*, 2004; Marcello and Toth, 1990). The exact algorithms, however, are typically expensive in terms of computational complexity, and thus, are not suitable to be used within the budget–limited MAB, since they may fail to fulfil our second research requirement (computational feasibility). Given this, we do not apply the abovementioned exact algorithms within the budget–limited bandit domain.

Another way to tackle the unbounded knapsack problem is to provide approximation algorithms (Kellerer *et al.*, 2004; Marcello and Toth, 1990). One of the simplest approximation approaches is to relax the unbounded knapsack problem so that the value of x_i can be fractional, instead of integers. This relaxation is referred to as the *fractional unbounded knapsack*, or the linear programming relaxation of the problem (Kellerer *et al.*, 2004). In more detail, the fractional unbounded knapsack can be formalised as follows:

$$\text{maximise} \quad \sum_{i=1}^{K} x_i v_i, \tag{2.27}$$

$$\text{subject to} \quad \sum_{i=1}^{K} x_i w_i \leq C, \tag{2.28}$$

$$\forall i \in \{1, \dots, K\} : \; x_i \geq 0. \tag{2.29}$$

To solve the fractional unbounded knapsack, we first define some notation. We refer to the fraction $\frac{v_i}{w_i}$ as the *density* of item type i. Let I^* denote the item type with the

Algorithm 2.2 Density–Ordered Greedy Algorithm

1: $N(1) = \{1, \ldots, K\}$, $t = 1$;
2: **while** packing is feasible **do**
3: $I^*(t) = \arg\max_{j \in N(t)} \{\frac{v_j}{w_j}\}$;
4: **while** packing is feasible **do**
5: pack item from type $I^*(t)$;
6: **end while**
7: $N(t+1) = N(t) \setminus \{I^*(t)\}$;
8: $t = t + 1$;
9: **end while**

highest density. That is, we have:

$$I^* = \arg\max_i \{\frac{v_i}{w_i}\}$$

It is easy to show that the optimal solution vector $\mathbf{x}^{\text{fr}} = \langle x_1^{\text{fr}}, \ldots, x_K^{\text{fr}} \rangle$ of the fractional relaxation problem is given by:

$$x_{I^*}^{\text{fr}} = \frac{C}{w_{I^*}},$$
$$x_j^{\text{fr}} = 0, \text{ for } j \neq I^*, \ j = 1, \ldots, K.$$

That is, the optimal solution is to solely use item type I^*, and ignore the others. Given this, the fractional knapsack based approximation algorithm is to *repeatedly pack items from item type* I^* into the knapsack (see Algorithm 2.1). It is easy to show that this algorithm has an approximation factor of $\frac{1}{2}$, and this factor is tight. Besides, the computational complexity of the algorithm is $O(K)$, since we just need to determine the type with the highest density (Dantzig, 1957; Kellerer *et al.*, 2004).

More recently, Kohli *et al.* (2004) studied an advanced version of the fractional knapsack relaxation, called the *density–ordered greedy* algorithm. In particular, the algorithm can be described as follows. First, the item types are sorted in order of their density, which is an operation of $O(K \log K)$ computational complexity, where K is the number of item types. Next, in the first round of this algorithm, as many units of the highest density item are selected as is feasible without exceeding the knapsack capacity. Then, in the second round, the densest item of the remaining feasible items is identified, and as many units of it as possible are selected. This step is repeated until there are no feasible items left (see Algorithm 2.2). Clearly, the maximum number of rounds is K. That is, the total computational cost is $O(K \log K + K)$. Note that the algorithm was also studied by Dantzig (1957). However, Kohli *et al.* (2004) improved the performance analysis of the algorithm, by showing that the approximation factor of the density–ordered greedy is $\frac{2}{3}$ *on average*, while Dantzig provided a tight approximation factor of $\frac{1}{2}$ for the worst–case performance.

Both the fractional relaxation based and the density–ordered greedy algorithms are simple in terms of computational cost, but they still achieve high performance in experiments, compared to the computationally expensive exact algorithms (Kellerer *et al.*, 2004; Kohli *et al.*, 2004; Marcello and Toth, 1990; Pisinger, 2005). Given this, we chose the fractional relaxation based and the density–ordered greedy methods as the foundations of our algorithms within the subsequent chapters. Note that there exist other, more sophisticated, approximation algorithms, that also achieve high performance in terms of low computational complexity (using the FPTAS framework). This includes, but is not limited to, the following works: Ibarra and Kim (1975); Lawler (1979), Marcello and Toth (1990), and Kellerer and Pferschy (1999). These algorithms, however, are significantly more expensive in terms of computational complexity, compared to the fractional relaxation based and the density–ordered greedy algorithms (for more details see (Kellerer *et al.*, 2004)). Thus, we ignore these algorithms within this thesis.

2.6 Summary

Within this chapter, we reviewed the literature of relevance in the topics of multi–armed bandits and knapsack problems. In particular, we described the standard stochastic MAB, which forms the basis of all the bandit models. Following this, we discussed the state–of–the–art stochastic bandit pulling policies. These approaches can typically be grouped into the following three classes: greedy based, UCB based, and probability matching. We pointed out that the greedy based algorithms typically outperform the others in applications, but they do not have strong theoretical performance guarantees. On the other hand, the more sophisticated UCB based algorithms are theoretically strong (i.e. they have efficient performance guarantees), but are typically outperformed by simpler algorithms such as ε–first or ε–greedy, especially in large problem settings (i.e. problems with a large number of arms). Meanwhile, the probability matching approaches are neither good in applications (outperformed by the greedy methods) or in theory (outperformed by the UCB based algorithms). We also pointed out the decreasing ε–greedy algorithm is a good candidate to be an efficient trade–off between providing good performance from both theoretical and experimental aspects. However, none of these algorithms take pulling cost into account, and thus, they are not suitable for the budget–limited multi–armed bandit problem. Nevertheless, they form a solid basis to our approaches in the subsequent chapters. In more detail, we combine ε–first, UCB and decreasing ε–greedy algorithms with unbounded knapsack techniques to tackle the budget–limited MAB problem.

We continued the literature review with the discussion of existing variants of the bandit model. We divided these models into four groups, based on the perspective from which they divert from the standard model. These perspectives are the following: (i) set of

arms; (ii) nature of rewards; (iii) additional information; and (iv) pulling costs. We demonstrated that among these variants, the bandit models with pulling costs are most related to our bandit setting. In particular, they also consider the budget limit and the pulling costs. However, they typically focus on different objectives, such as minimising the switching cost, the simple regret, or the active learning regret. Given this, we pointed out that the pulling policies, that are designed to tackle these bandit problems, are not likely to provide good performance in our budget–limited MAB. However, we can extend the work within this thesis by taking pulling costs into consideration in many of these models, and thus they form the basis of possible future work (as elaborated upon in Section 8.2).

Following this, we turned to the discussion of the knapsack problems. In more detail, we described the standard knapsack model and a number of its variants, including the unbounded knapsack. We next focused on the exact and approximation algorithms of the unbounded version. In particular, we pointed out that the exact algorithms are computationally expensive, and thus, they cannot fulfil our second research requirement (see Section 1.1 for more details). We focus on two simple, but efficient approximation methods, namely the fractional unbounded knapsack based and the density–ordered greedy algorithms. These methods form the foundation of our solutions in order to tackle the budget–limited MAB. In particular, within the subsequent chapters, we will show that the the budget–limited MAB can be efficiently tackled by combining fractional unbounded knapsack based and the density–ordered greedy algorithms with the ε–first, UCB and decreasing ε–greedy methods from the stochastic MAB domain.

In summary, we showed that to date, none of the state–of–the–art studies has addressed the problem of budget–limited multi–armed bandits, and thus, no efficient pulling policies have been made within this bandit setting. Against this background, one of the main drives of our work is to fill this gap, by designing efficient pulling algorithms that satisfy our research requirements. In particular, our contributions can be distinguished into: (ii) ε–first based approach, that is efficient in experimental applications, but has weak theoretical bounds (see Chapter 4); (ii) two UCB based approaches, that are efficient in theory, but provide poor performance in experiments (see Chapter 5); (iii) two ε–greedy based algorithms, that provide a trade–off between the aforementioned approaches (see Chapter 6). In addition, we will demonstrate the usefulness of the budget–limited MAB in Chapter 7, where we apply our bandit model to the problem of long–term information collection of wireless sensor networks.

Chapter 3

Formal Description of Budget–Limited Multi–Armed Bandits

Given the description of research objectives and literature of relevance in the previous chapters, we now formalise the budget–limited multi–armed bandit problem. The budget–limited MAB model consists of a slot machine with K arms, one of which must be pulled by the agent at each time step. By pulling arm i, the agent has to pay a pulling cost, denoted with c_i, and receives a non–negative reward drawn from a distribution associated with that specific arm. The agent has a cost budget B, which it cannot exceed during its operation time (i.e. the total cost of pulling arms cannot exceed this budget limit). Now, since reward values are typically bounded in real–world applications, we assume that each arm's reward distribution has bounded supports. Without loss of generality, for ease of exposition we assume that the reward distribution of each arm has support in $[0, 1]$, and that the pulling cost $c_i \geq 1$ for each i (our result can be scaled for different size supports and costs as appropriate). Let μ_i denote the mean value of the rewards that the agent receives from pulling arm i. Within our model, the agent's goal is to maximise the sum of rewards it earns from pulling the arms of the machine, with respect to the budget B. However, the agent has no initial knowledge of the μ_i of each arm i, so it must learn these values in order to deduce a policy that maximises its sum of rewards. Given this, our objective is to find the optimal pulling algorithm, which maximises the expectation of the total reward that the agent can achieve, without exceeding the cost budget B.

Formally, let A be an arm–pulling algorithm, giving a finite sequence of pulls. Let $N_i^B(A)$ be the random variable that denotes the number of pulls of arm i by A, with respect to the budget limit B. Note that the total cost of the sequence A cannot exceed

B, that is:

$$P\left(\sum_{i}^{K} N_i^B(A)\, c_i \leq B\right) = 1. \tag{3.1}$$

Let $G^B(A)$ be the total reward earned by using A to pull the arms with respect to budget limit B. The expectation of $G^B(A)$ is:

$$\mathbf{E}\left[G^B(A)\right] = \sum_{i}^{K} \mathbf{E}\left[N_i^B(A)\right] \mu_i. \tag{3.2}$$

Then, let A^* denote an optimal solution that maximises the expected total reward, that is:

$$A^* = \arg\max_{A} \sum_{i}^{K} \mathbf{E}\left[N_i^B(A)\right] \mu_i. \tag{3.3}$$

Note that in order to determine A^*, we have to know the value of μ_i in advance, which does not hold in our case. Thus, A^* represents a theoretical optimum value, which is unachievable in general.

Nevertheless, for any algorithm A, we can define the regret for A as the difference between the expected cumulative reward for A and that of the theoretical optimum A^*. More precisely, letting $R_B(A)$ denote the regret, we have:

$$R^B(A) = \mathbf{E}\left[G^B(A^*)\right] - \mathbf{E}\left[G^B(A)\right]. \tag{3.4}$$

Our objective is to derive a method of generating a sequence of arm pulls that minimises this regret for the class of MAB problems defined above. In so doing, we define some useful terms, that can be formalised as follows. Let I^* denote the arm with the *highest reward mean density*, that is:

$$I^* = \arg\max_{i} \frac{\mu_i}{c_i}. \tag{3.5}$$

For the sake of simplicity, we assume that I^* is unique. However, this assumption does not put restriction on any of our results. For each sub–optimal arm i (i.e. $i \neq I^*$), we define d_i as the difference between the reward mean density of I^* and that of i:

$$d_i = \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_i}{c_i}. \tag{3.6}$$

Let $d_{\min}$ denote the minimum value of these:

$$d_{\min} = \min_{d_i > 0} d_i = \min_{i \neq I^*} \left\{ \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_i}{c_i} \right\}, \tag{3.7}$$

In addition, for each sub–optimal arm i, let

$$\Delta_i = \mu_{I^*} - \mu_i, \tag{3.8}$$

$$\delta_i = c_i - c_{I^*}. \tag{3.9}$$

Note that Δ_i or δ_i can be negative, since it is possible that $\mu_{I^*} < \mu_i$, or $c_{I^*} > c_i$. However, it is easy to show that both of Δ_i and δ_i cannot be negative at the same time, since $\frac{\mu_{I^*}}{c_{I^*}} \geq \frac{\mu_i}{c_i}$ for all i.

Finally, let $c_{\min}$ and $c_{\max}$ denote the lowest and largest pulling cost, respectively. That is, we get:

$$c_{\min} = \min_i c_i, \tag{3.10}$$

$$c_{\max} = \max_i c_i. \tag{3.11}$$

Using the formalisations described above, in what follows, we propose three classes of algorithms to tackle the budget–limited MAB:

- an ε–first based approach (Chapter 4),
- two UCB based approaches (Chapter 5),
- and two ε–greedy based algorithms (Chapter 6).

These approaches are described in more detail in the subsequent chapters. In particular, Chapters 4, 5, and 6 focus on the algorithms' fulfilment of Requirements 2 (computational feasibility), and 3 (efficient finite–time regret bound), respectively. In addition, we study the empirical efficiency of the algorithms in Chapter 7, in order to analyse their fulfilment in Requirement 1 (empirical performance quality).

Chapter 4

Budget–Limited Epsilon–First based Approaches

Having devised a model for budget–limited multi–armed bandits, we now outline a number of pulling algorithms to tackle this bandit problem. In this chapter, we concentrate on the budget–limited ε–first approach, in which the first ε of the overall budget B is dedicated to exploration, and the remaining portion is dedicated to exploitation. To this end, in Section 4.1, we describe the algorithm in more detail. This is followed by a performance analysis in Section 4.2. In particular, we study the computational complexity, and we propose theoretical upper bounds for the performance regret of the approach.

4.1 The Algorithm

In the budget–limited ε–first approach, we first purely explore until we exceed the exploration budget εB, then we estimate the best combination of arms, based on the estimated values of the rewards (see Algorithm 4.1), and then repeatedly pull this combination. Here, let t denote the time step, and B_t^{expl} denote the residual exploration budget at time t, respectively. Note that at the start (i.e. $t = 1$), $B_1^{\text{expl}} = \varepsilon B$, where B is the total budget limit.

In more detail, within the exploration phase, we uniformly pull the arms, with respect to the exploration budget εB. That is, we sequentially pull all of the arms, one after the other, until the exploration budget is exceeded (steps $3 - 9$). In particular, at time step t, we pull arm $i\,(t) = t \mod K$ (step 7). Note that since there is no arm 0, we denote $mK \mod K = K$ for any integer m (i.e. we replace the congruency class 0 with class K). The reason of choosing this method is that, in order to bound the regret of the algorithm, since we do not know which arms will be pulled in the exploitation phase, we need to treat the arms equally in the exploration phase.

Algorithm 4.1 Budget–Limited ε–First Algorithm

Exploration phase:
$t = 1$; $B_t^{\text{expl}} = \varepsilon B$;
while pulling is feasible **do**
  if $B_t^{\text{explore}} < \min_i c_i$ **then**
    STOP! {pulling is not feasible}
  end if
  pull arm $i(t)$, where $i(t) = t \mod K$ {choose the subsequent arm to pull};
  $B_{t+1}^{\text{expl}} = B_t^{\text{expl}} - c_{i(t)}$; $t = t + 1$;
end while
Exploitation phase:
use density–ordered greedy to pull the arms;

Following this, we focus on a pure exploitation phase. In so doing, we reduce the problem faced by an agent in the exploitation phase to the unbounded knapsack problem (see Section 2.5). Recall that in the exploitation phase, the agent makes use of the expected reward estimates from the exploration phase, which can be calculated as follows. Suppose that the exploration phase stops after T steps. Let $r(t)$ denote the reward received by pulling arm $i(t)$ at time step t (step 7). Let n_i denote the number of times the agent pulls arm i until T. We define $\hat{\mu}_{i,n_i}$ as the estimate of μ_i after the exploration phase, which can be calculated as follows:

$$\hat{\mu}_{i,n_i} = \frac{1}{n_i} \sum_{t=1}^{T} \mathbf{I}_{\{i=t \mod K\}} r(t), \tag{4.1}$$

where $\mathbf{I}_{\{i=t \mod K\}}$ is the indicator function of the event $\{i = t \mod K\}$. That is, $\hat{\mu}_{i,n_i}$ is the average of rewards the agent receives by pulling arm i during the exploration phase. Given this we aim to solve the following unbounded knapsack:

$$\max \sum_{i=1}^{k} x_i \hat{\mu}_{i,n_i} \quad \text{s.t.} \quad \sum_{i=1}^{k} x_i c_i \leq (1-\varepsilon) B,$$

where x_i is the number of pulls of arm i in the exploitation phase. In this case, the ratio of an arm's reward estimate to its pulling cost, $\frac{\hat{\mu}_{i,n_i}}{c_i}$, is analogous to the "density" of an item, because it represents the reward for consuming one unit of the budget, or one unit of the carrying capacity of the knapsack. As such, the problem is equivalent to the knapsack problem above, and in order to solve it, we can use a density–ordered greedy algorithm at step 11 (see Section 2.5.2 for more details).

Intuitively, this approach is motivated by the fact that the theoretical optimal solution of the budget–limited MAB is a combination of pulls that might contain a variety of different arms (see Chapter 1 for more detail). Thus, by estimating the expected reward value of all the arms in the exploration phase, the budget–limited ε–first approach can efficiently estimate the optimal combination of pulls within the exploitation phase. In more detail, by explicitly splitting exploration from exploitation, we can easily measure the accuracy of the estimates associated with a particular value of ε, because all of the arms are sampled the same number of times. Hence, we can control the performance regret as a function of ε, which gives us a method of choosing an optimal ε for a given scenario.

4.2 Performance Analysis

We now turn to the performance analysis of the budget–limited ε–first approach. In particular, we first derive a linear upper bound for the performance regret of budget–limited ε–first. This bound, however, does not satisfy Requirement 3; that is, it does not follow the concept of asymptotic optimal convergence (see Section 1.1 for more details). Following this, we improve this result by providing a probably approximately correct (PAC) regret bound (see Section 2.2) for any exploration policy and the density–ordered greedy algorithm (i.e. the upper bound is independent of the choice of the exploration algorithm). In particular, a PAC bound holds with a certain probability, while it might be violated in a small amount of cases. We then refine this bound for the specific case of uniform pull exploration, and we show that by optimally tuning the value of ε, the PAC regret bound is improved to be $O\left(B^{\frac{2}{3}}\right)$. We show that the improved result guarantees Requirement 3 within the PAC manner (i.e. it holds with high probability). Finally, we study the computational cost of the approach, in order to verify whether the budget–limited ε–first approach satisfies Requirement 2.

Our first result regarding the performance regret of the approach is described as follows:

Theorem 4.1. *For any budget size $B > 0$, the performance regret of the budget–limited ε–first approach is at most*

$$\varepsilon B \left(\frac{\mu_{I^*}}{c_{I^*}} - \frac{\sum_j^K \mu_j}{\sum_j^K c_j} \right) + 2\left(1-\varepsilon\right) B \sum_{j \neq I^*} d_j \exp\left\{ -\frac{c_{\min}^2 d_{\min}^2}{2\sum_j^K c_j} \varepsilon B \right\} \exp\left\{ \frac{c_{\min}^2 d_{\min}^2}{2} \right\} + K + 1,$$

where $d_j = \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j}$ for earch arm j.

To prove this theorem, recall that $\hat{\mu}_{i,n_i}$ denotes the estimated value of the expected reward of arm i, where n_i is the number of pulls of that arm in the exploration phase. Hereafter, for the sake of simplicity, we refer to $\hat{\mu}_{i,n_i}$ as $\hat{\mu}_i$ (i.e. we leave n_i from the subscript). Let A_{uniform} denote the the uniform pull exploration policy. In addition, let A_{arb} denote an arbitrary exploration policy (which can be uniform as well), and A_{greedy} denote the density–ordered greedy exploitation algorithm, respectively.

Within this section (and in the subsequent chapters as well), we will make use of the following version of the Chernoff–Hoeffding concentration inequality for bounded random variables:

Theorem 4.2 (Chernoff–Hoeffding inequality (Hoeffding, 1963))**.** *Let $X_1, X_2, \ldots, X_n$ denote the sequence of random variables with common range $[0,1]$, such that for any $1 \leq t \leq n$, we have $\mathbf{E}\left[X_t | X_1, \ldots, X_{t-1}\right] = \mu$. Let $S_n = \frac{1}{n}\sum_{t=1}^n X_t$. Given this, for any*

$\delta \geq 0$, we have:

$$P\left(S_n \geq \mu + \delta\right) \leq e^{-2n\delta^2}, \tag{4.2}$$

$$P\left(S_n \leq \mu - \delta\right) \leq e^{-2n\delta^2}. \tag{4.3}$$

The proof can be found, for example, in Hoeffding (1963). Using this, we prove Theorem 4.1 as follows:

Proof of Theorem 4.1. To estimate the regret of the budget–limited ε–first approach, we separately estimate the regret of the uniform exploration and density–ordered greedy exploitation policies. In particular, recall that A_{uniform} sequentially pulls each arm i until it exceeds the exploration budget εB. That is, the expected total reward that the agent receives with this exploration policy is

$$\mathbf{E}\left[G^{\varepsilon B}\left(A_{\text{uniform}}\right)\right] = \sum_{i=1}^{K} n_i \mu_i. \tag{4.4}$$

It is easy to show that for each arm i:

$$\left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor \leq n_i \leq \left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor + \frac{\sum_{j=1}^{K} c_j}{c_{\min}}, \tag{4.5}$$

where n_i denotes the number of times A_{uniform} pulls arm i. Using Equation 4.5, we have:

$$\begin{aligned} \mathbf{E}\left[G^{\varepsilon B}\left(A_{\text{uniform}}\right)\right] &\geq \sum_{i=1}^{K} \mu_i \left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor \\ &\geq \sum_{i=1}^{K} \mu_i \left(\frac{\varepsilon B}{\sum_{j=1}^{K} c_j} - 1 \right) \\ &\geq \sum_{i=1}^{K} \mu_i \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} - K. \end{aligned} \tag{4.6}$$

The last inequality is obtained from the fact that $0 \leq \mu_i \leq 1$ for all arms i (see Chapter 3 for more details). Now, it is easy to show that within the exploration phase, the theoretical optimal total expected reward that any policy can receive is $\frac{\varepsilon B}{c_{I^*}} \mu_{I^*}$; that is, by repeatedly pull the arm with the best expected reward density. This implies the following:

$$R^{\varepsilon B}\left(A_{\text{uniform}}\right) \leq \frac{\varepsilon B}{c_{I^*}} \mu_{I^*} - \sum_{i=1}^{K} \mu_i \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} - K = \varepsilon B \left(\frac{\mu_{I^*}}{c_{I^*}} - \frac{\sum_{j=1}^{K} \mu_j}{\sum_{j=1}^{K} c_j} \right) + K. \tag{4.7}$$

Now we turn to estimate $R^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)$ as follows. Recall that A_{greedy} first repeatedly pulls I^+, the arm with the highest estimated expected reward density after exploration,

until it is not feasible with respect to the residual budget $(1-\varepsilon)B$. Thus, we have:

$$\begin{aligned}\mathbf{E}\left[G^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)|I^{+}\right] &\geq \left\lfloor\frac{(1-\varepsilon)B}{c_{I^{+}}}\right\rfloor\mu_{I^{+}}\\ &\geq \left(\frac{(1-\varepsilon)B}{c_{I^{+}}}-1\right)\mu_{I^{+}}\\ &\geq (1-\varepsilon)B\frac{\mu_{I^{+}}}{c_{I^{+}}}-1. \end{aligned} \tag{4.8}$$

Similar to the exploration phase, the theoretical optimal total expected reward that any policy can receive within the exploitation phase is to repeatedly pulling the arm with the best expected reward density; that is, $\frac{(1-\varepsilon)B}{c_{I^{*}}}\mu_{I^{*}}$. This implies that for a particular I^{+}, we have:

$$R^{(1-\varepsilon)B}\left(A_{\text{greedy}}|I^{+}\right)\leq(1-\varepsilon)B\left(\frac{\mu_{I^{*}}}{c_{I^{*}}}-\frac{\mu_{I^{+}}}{c_{I^{+}}}\right)+1, \tag{4.9}$$

where $R^{(1-\varepsilon)B}\left(A_{\text{greedy}}|I^{+}\right)$ denotes the regret of A_{greedy} conditional to I^{+}. By summing up over all the possible values of I^{+}, we get:

$$R^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)\leq\sum_{j=1}^{K}\left((1-\varepsilon)Bd_{j}+1\right)P\left(I^{+}=j\right).$$

The right hand side can be reformalised as:

$$R^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)\leq(1-\varepsilon)B\sum_{j\neq I^{*}}d_{j}P\left(I^{+}=j,I^{*}\neq j\right)+1. \tag{4.10}$$

Here, $P\left(I^{+}=j,I^{*}\neq j\right)$ denotes the probability that the arm with the best estimated expected reward density is not equal to I^{*}. In what follows, we provide an upper bound for $P\left(I^{+}=j,I^{*}\neq j\right)$, in order to estimate Equation 4.10. Note that the following holds:

$$P\left(I^{+}=j,I^{*}\neq j\right)\leq P\left(\frac{\hat{\mu}_{j}}{c_{j}}\geq\frac{\hat{\mu}_{I^{*}}}{c_{I^{*}}}\right).$$

This can be further bound by noting the following: Let $X,Y\in\mathbb{R}$ be independent random variables, and $c\in\mathbb{R}$. Thus, $P\left(X\geq Y\right)\leq P\left(X\geq c\right)+P\left(Y\leq c\right)$, because

$$\begin{aligned}P\left(X\geq Y\right) &\leq P\left(X\geq Y|X\geq c\right)+P\left(X\geq Y|X\leq c\right)\\ &\leq P\left(X\geq c\right)+P\left(Y\leq c|X\leq c\right)=P\left(X\geq c\right)+P\left(Y\leq c\right).\end{aligned}$$

Given this, we have:

$$P\left(I^{+}=j,I^{*}\neq j\right)\leq P\left(\frac{\hat{\mu}_{j}}{c_{j}}\geq\frac{\mu_{j}}{c_{j}}+\frac{d_{j}}{2}\right)+P\left(\frac{\hat{\mu}_{I^{*}}}{c_{I^{*}}}\leq\frac{\mu_{I^{*}}}{c_{I^{*}}}-\frac{d_{j}}{2}\right). \tag{4.11}$$

Note that by definition of d_j, we have $\frac{\mu_j}{c_j} + \frac{d_j}{2} = \frac{\mu_{I^*}}{c_{I^*}} - \frac{d_j}{2}$. Using the Chernoff–Hoeffding inequality for both terms on the right hand side of Equation 4.11, we get:

$$\begin{aligned} P\left(\frac{\hat{\mu}_j}{c_j} \geq \frac{\mu_j}{c_j} + \frac{d_j}{2}\right) &= P\left(\hat{\mu}_j \geq \mu_j + \frac{c_j d_j}{2}\right) \\ &\leq \exp\left\{-\frac{c_j^2 d_j^2 n_j}{2}\right\}. \end{aligned} \tag{4.12}$$

where n_j denotes the number of times A_{uniform} (i.e. the uniform exploration policy) pulls arm j. From Equation 4.5 we can show that $n_j \geq \frac{\varepsilon B}{c_j} - 1$. Combining this with Equation 4.12, we obtain:

$$P\left(\frac{\hat{\mu}_j}{c_j} \geq \frac{\mu_j}{c_j} + \frac{d_j}{2}\right) \leq \exp\left\{-\frac{c_j^2 d_j^2 \varepsilon B}{2\sum_{i=1}^{K} c_i}\right\} \exp\left\{\frac{c_j^2 d_j^2}{2}\right\}. \tag{4.13}$$

In a similar vein, we can show that:

$$P\left(\frac{\hat{\mu}_{I^*}}{c_{I^*}} \leq \frac{\mu_{I^*}}{c_{I^*}} - \frac{d_j}{2}\right) \leq \exp\left\{-\frac{c_j^2 d_j^2 \varepsilon B}{2\sum_{i=1}^{K} c_i}\right\} \exp\left\{\frac{c_j^2 d_j^2}{2}\right\}. \tag{4.14}$$

Substituting Equations 4.13 and 4.14 into Equation 4.11 we get:

$$P\left(I^+ = j, I^* \neq j\right) \leq 2\exp\left\{-\frac{c_j^2 d_j^2 \varepsilon B}{2\sum_{i=1}^{K} c_i}\right\} \exp\left\{\frac{c_j^2 d_j^2}{2}\right\}. \tag{4.15}$$

This implies that:

$$R^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right) \leq 2\left(1-\varepsilon\right) B \sum_{j \neq I^*} d_j \exp\left\{-\frac{c_j^2 d_j^2 \varepsilon B}{2\sum_{i=1}^{K} c_i}\right\} \exp\left\{\frac{c_j^2 d_j^2}{2}\right\} + 1. \tag{4.16}$$

Since $R^B\left(\varepsilon\text{−first}\right) = R^{\varepsilon B}\left(A_{\text{uniform}}\right) + R^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)$, we conclude the proof by adding Equations 4.7 and 4.16 together. □

Note that this bound depends on the value of ε. Thus, we can further improve the bound by choosing an optimal ε value. However, by using elementary techniques, it can be easily to proven that the optimal value of ε that minimises the equation in Theorem 4.1 is either $\varepsilon = 0$ or $\varepsilon = 1$. In both cases, we can see that the regret bound is $O\left(B\right)$ (i.e. a linear function of budget B). That is, the upper bound of the budget–limited ε–first approach given in Theorem 4.1 can be improved to be $O\left(B\right)$ in the best case. This implies that the regret bound is in fact not efficient. In more detail, it can be easily shown that this bound does not follow the concept of optimal asymptotic convergence; that is, it does not guarantee that the average regret converges to 0 with probability 1 as the number of time steps tends to infinity. Given this, this regret bound does not meet Requirement 3.

Nevertheless, we can improve the regret bound if we allow the bound to be violated in a small number of cases. Thus, in what follows, we focus on PAC (probably approximately correct) type bounds. Let I^+ denote the arm with the highest estimated density after the exploration phase:

$$I^+ = \arg\max_j \left\{ \frac{\hat{\mu}_{j,n_j}}{c_j} \right\}. \tag{4.17}$$

In addition, we define $d_{\max}$ as follows:

$$d_{\max} = \max_j \left\{ \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} \right\}. \tag{4.18}$$

That is, $d_{\max}$ denotes the largest difference between the expected reward density of the arms. Finally, we say that an exploration policy exploits the budget dedicated to the exploration phase if and only if after the exploration stops, none of the arms can be additionally pulled without exceeding the exploration budget. As a result, we have the following:

Theorem 4.3. *Consider a budget–limited ε–first approach with an arbitrary exploration policy that exploits the exploration budget. In addition, suppose that all the arms are pulled at least once within this exploration phase. For any $B > 0$, and $0 < \varepsilon, \beta < 1$, with at least $(1-\beta)^K$ probability, the performance regret of the budget–limited ε–first approach is at most*

$$2 + \varepsilon B d_{\max} + B \left(\sqrt{\frac{-\ln \frac{\beta}{2}}{2n_{I^*}}} + \sqrt{\frac{-\ln \frac{\beta}{2}}{2n_{I^+}}} \right),$$

where n_{I^*} and n_{I^+} are the number of pulls of arms I^* and I^+ within the exploration phase, respectively.

To prove this theorem, we define $I^{\min}$ as the arm with minimal expected reward density. That is,

$$I^{\min} = \arg\min_j \frac{\mu_j}{c_j}. \tag{4.19}$$

We rely on the following auxiliary lemmas:

Lemma 4.4. *Suppose that A_{arb} is an arbitrary exploration policy that exploits its exploration budget. Within this policy, each arm i is pulled n_i times. Thus, we have*

$$\sum_{i=1}^{k} n_i \mu_i \geq \frac{\varepsilon B \mu_{I^{\min}}}{c_{I^{\min}}} - 1.$$

Lemma 4.5. *For the density-ordered greedy exploitation algorithm A_{greedy}, we have:*

$$\mathbf{E}\left[G^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right) \right] \geq (1-\varepsilon)\frac{B\mu_{I^+}}{c_{I^+}} - 1.$$

Lemma 4.6. *If A^* is the optimal solution of the budget–limited MAB, then*

$$\mathbf{E}\left[G^B\left(A^*\right)\right] \leq \frac{B\mu_{I^*}}{c_{I^*}}.$$

Lemma 4.7. *If $|a-b| \leq \delta_1$, $|c-d| \leq \delta_2$, $a \geq c$, then $d \leq b + \delta_1 + \delta_2$.*

Proof of Lemma 4.4. If A_{arb} exploits the budget for exploration, it is true that for any c_j:

$$\sum_{i=1}^{K} n_i c_i \geq \varepsilon B - c_j$$

since none of the arms can be pulled after the stop of A_{arb}, without exceeding εB. Furthermore, $\mu_i = c_i\left(\frac{\mu_i}{c_i}\right) \geq c_i\left(\frac{\mu_{I\min}}{c_{I\min}}\right)$. Since $\mu_i \leq 1$, we have:

$$\sum_{i=1}^{K} n_i\mu_i \geq \left(\sum_{i=1}^{K} n_i c_i\right)\frac{\mu_{I^{\min}}}{c_{I^{\min}}} \geq \left(\varepsilon B - c_{I^{\min}}\right)\frac{\mu_{I^{\min}}}{c_{I^{\min}}} \geq \frac{\varepsilon B\mu_{I^{\min}}}{c_{I^{\min}}} - 1.$$

□

Proof of Lemma 4.5. By just pulling arm I^+ in the exploitation phase, which is the first round of A_{greedy}, the expected reward we can get there is $\left\lfloor\frac{(1-\varepsilon)B}{c_{I^+}}\right\rfloor\mu_{I^+}$. Since $\left\lfloor\frac{(1-\varepsilon)B}{c_{I^+}}\right\rfloor > \left(\frac{(1-\varepsilon)B}{c_{I^+}} - 1\right)$, we have:

$$\mathbf{E}\left[G^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)\right] \;\geq\; \left(\frac{(1-\varepsilon)\,B}{c_{I^+}} - 1\right)\mu_{I^+} \geq (1-\varepsilon)\,\frac{B\mu_{I^+}}{c_{I^+}} - 1,$$

since $\mu_i \leq 1$ for $\forall i$. □

Proof of Lemma 4.6. Suppose that in the optimal solution, N_i is the total number of pulls of arm i. Thus, the cost constraint can be formulated as:

$$\sum_{i=1}^{K} N_i c_i \leq B.$$

Given this, we have:

$$\mathbf{E}\left[G^B\left(A^*\right)\right] = \sum_{i=1}^{k} N_i\mu_i = \sum_{i=1}^{k} N_i c_i\frac{\mu_i}{c_i} \leq \left(\sum_{i=1}^{k} N_i c_i\right)\frac{\mu_{I^*}}{c_{I^*}} \leq \frac{B\mu_{I^*}}{c_{I^*}}.$$

□

Proof of Lemma 4.7. Since $|a-b| \leq \delta_1$, we have $a \leq b+\delta_1$. Similarly, we have $d \leq c+\delta_2$. Since $a \geq c$, we have the following: $d \leq c+\delta_2 \leq a+\delta_2 \leq b+\delta_1+\delta_2$. □

Given the aforementioned lemmas, we now turn to prove Theorem 4.3 as follows:

Proof of Theorem 4.3. Using the Chernoff–Hoeffding inequality for each arm i, and for any positive δ_i, we have:

$$P\left(|\hat{\mu}_i - \mu_i| \geq \delta_i\right) \leq 2\exp\left\{-2n_i\delta_i^2\right\},$$

that is, dividing by c_i, we have:

$$P\left(\left|\frac{\hat{\mu}_i}{c_i} - \frac{\mu_i}{c_i}\right| \geq \frac{\delta_i}{c_i}\right) \leq 2\exp\left\{-2n_i\delta_i^2\right\},$$

which is equivalent to the following:

$$P\left(\left|\frac{\hat{\mu}_i}{c_i} - \frac{\mu_i}{c_i}\right| \geq \delta_i\right) \leq 2\exp\left\{-2n_i\delta_i^2c_i^2\right\}. \tag{4.20}$$

By setting $\delta_i = \sqrt{\frac{-\ln\frac{\beta}{2}}{2n_ic_i^2}}$, Equation (4.20) can be reformulated as follows:

$$P\left(\left|\frac{\hat{\mu}_i}{c_i} - \frac{\mu_i}{c_i}\right| \geq \delta_i\right) \leq \beta.$$

Thus, with at least $(1-\beta)^K$ probability, for each arm i, we have

$$\left|\frac{\hat{\mu}_i}{c_i} - \frac{\mu_i}{c_i}\right| \leq \delta_i \tag{4.21}$$

holds for each arm i. Hereafter, we stricly focus on this case. Given this, the reward collected in the exploration phase can be calculated as follows:

$$\mathbf{E}\left[G^{\varepsilon B}\left(A_{\text{arb}}\right)\right] = \sum_{i=1}^{K} n_i\mu_i \geq \frac{\varepsilon B\mu_{I^{\min}}}{c_{I^{\min}}} - 1. \tag{4.22}$$

The right side of Equation 4.22 holds, due to Lemma 4.4. Using Lemma 4.5 and Equation 4.22, we get the following:

$$\begin{aligned}\mathbf{E}\left[G^{B}\left(\varepsilon\text{–first}\right)\right] &= \mathbf{E}\left[G^{\varepsilon B}\left(A_{\text{arb}}\right)\right] + \mathbf{E}\left[G^{(1-\varepsilon)B}\left(A_{\text{greedy}}\right)\right] \\ &\geq \frac{\varepsilon B\mu_{I^{\min}}}{c_{I^{\min}}} + (1-\varepsilon)\frac{B\mu_{I^+}}{c_{I^+}} - 2,\end{aligned} \tag{4.23}$$

where $G^B\left(\varepsilon-\text{first}\right)$ denotes the total reward that the budget–limited ε–first approach receives. By denifition, we have

$$\frac{\mu_{I^*}}{c_{I^*}} \geq \frac{\mu_{I^+}}{c_{I^+}},$$

and

$$\frac{\hat{\mu}_{I^+}}{c_{I^+}} \geq \frac{\hat{\mu}_{I^*}}{c_{I^*}}.$$

Furthermore, $\left|\frac{\hat{\mu}_i}{c_i} - \frac{\mu_i}{c_i}\right| \leq \delta_i$ holds for each arm i. Thus, according to Lemma 4.7, we have

$$\frac{\mu_{I^+}}{c_{I^+}} \geq \frac{\mu_{I^*}}{c_{I^*}} - \delta_{I^*} - \delta_{I^+}. \tag{4.24}$$

Substituting this into Equation 4.23, we have:

$$\begin{aligned}\mathbf{E}\left[G^B\left(\varepsilon-\text{first}\right)\right] \quad \geq \quad & \frac{\varepsilon B \mu_{I^{\min}}}{c_{I^{\min}}} + B\frac{\mu_{I^*}}{c_{I^*}} - \frac{\varepsilon B \mu_{I^*}}{c_{I^*}} - \\ & - (1-\varepsilon)\, B\left(\delta_{I^*} + \delta_{I^+}\right) - 2.\end{aligned} \tag{4.25}$$

According to Lemma 4.6, $\mathbf{E}\left[G^B\left(A^*\right)\right] \leq \frac{B\mu_{I^*}}{c_{I^*}}$. Thus, by substituting it into Equation 4.25, and using the definition of regret in Equation 3.4, we have:

$$R^B\left(\varepsilon-\text{first}\right) \leq 2 + \varepsilon B d_{\max} + B\left(\delta_{I^*} + \delta_{I^+}\right),$$

where $d_{\max} = \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_{I^{\min}}}{c_{I^{\min}}}$. Note that here we used the fact that $(1-\varepsilon) < 1$. Thus, by replacing $\delta_i = \sqrt{\frac{-\ln\frac{\beta}{2}}{2n_i c_i^2}}$ for $i = I^*$ and $i = I^+$, and using the fact that $c_i \geq 1$ for each i, we get the requested formula. □

Note that the aforementioned bound holds for any arbitrary exploration policy. We now refine this upper bound for the case of uniform pull exploration as follows:

Corollary 4.8. *Let $0 < \varepsilon, \beta < 1$. Suppose that $\varepsilon B \geq \sum_{j=1}^{K} c_j$. With probability $(1-\beta)^K$, the performance regret of the budget–limited ε–first approach with uniform pull exploration is at most*

$$2 + \varepsilon B d_{\max} + 2\sqrt{\frac{B\left(-\ln\frac{\beta}{2}\right)\sum_{j=1}^{K} c_j}{\varepsilon}}. \tag{4.26}$$

Proof. Recall that within A_{uniform} (i.e. the uniform exploration policy), for each arm i, we have:

$$\left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor \leq n_i \leq \left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor + \frac{\sum_{j=1}^{K} c_j}{c_{\min}}. \tag{4.27}$$

This implies that by using A_{uniform} in the budget–limited ε–first approach, we have:

$$R^B\left(\varepsilon\text{–first}\right) \leq 2 + \varepsilon B d_{\max} + 2B\left(\sqrt{\frac{-\ln\frac{\beta}{2}}{2\left\lfloor\frac{\varepsilon B}{\sum_{j=1}^{K} c_j}\right\rfloor}}\right). \tag{4.28}$$

Now, if $\varepsilon B \geq \sum_{j=1}^{K} c_j$, we can show that:

$$\left\lfloor\frac{\varepsilon B}{\sum_{j=1}^{K} c_j}\right\rfloor \geq \frac{\varepsilon B}{2\sum_{j=1}^{K} c_j}.$$

The proof is elementary, and thus, is omitted. Substituting this into Equation 4.28 results in the following:

$$R^B\left(\varepsilon\text{–first}\right) \leq 2 + \varepsilon B d_{\max} + 2\sqrt{\frac{B\left(-\ln\frac{\beta}{2}\right)\sum_{j=1}^{K} c_j}{\varepsilon}}, \tag{4.29}$$

which concludes the proof. □

Setting the value of ε to be the minimal point of Equation 4.26 (i.e. the point that minimises this equation) implies the following result:

Corollary 4.9. *For any* $0 < \beta < 1$*, suppose that* $B \geq \sum_{j=1}^{K} c_j \max\left\{\frac{\sqrt{2\left(-\ln\frac{\beta}{2}\right)}}{d_{\max}}, \frac{2\left(-\ln\frac{\beta}{2}\right)}{d_{\max}^2}\right\}$. *With probability* $(1-\beta)^K$*, the performance regret of the budget–limited* ε*–first approach with uniform pull exploration is at most*

$$2 + 3B^{\frac{2}{3}}\left(\left(-\ln\frac{\beta}{2}\right)\sum_{j=1}^{K} c_j d_{\max}\right)^{\frac{1}{3}}$$

if the value of ε *is set to be:*

$$\varepsilon = \left(\frac{\left(-\ln\frac{\beta}{2}\right)\sum_{j=1}^{K} c_j}{B d_{\max}^2}\right)^{\frac{1}{3}} \leq 1.$$

Proof. It is easy to see that if we consider Equation 4.26 as a function of ε, then the global minimum point is set at

$$\varepsilon_{\text{opt}} = \left(\frac{2\left(-\ln\frac{\beta}{2}\right)\sum_{j=1}^{K} c_j}{B d_{\max}^2}\right)^{\frac{1}{3}}.$$

Note that we have to guarantee that both $\varepsilon_{\text{opt}} \leq 1$ and $\varepsilon_{\text{opt}} B \geq \sum_{j=1}^{K} c_j$ hold. The former holds if $B \geq \sum_{j=1}^{K} c_j \frac{2\left(-\ln \frac{\beta}{2}\right)}{d_{\max}^2}$, and the latter holds if $B \geq \sum_{j=1}^{K} c_j \frac{\sqrt{2\left(-\ln \frac{\beta}{2}\right)}}{d_{\max}}$. Thus, by setting $B \geq \sum_{j=1}^{K} c_j \max\left\{\frac{\sqrt{2\left(-\ln \frac{\beta}{2}\right)}}{d_{\max}}, \frac{2\left(-\ln \frac{\beta}{2}\right)}{d_{\max}^2}\right\}$, both $\varepsilon_{\text{opt}} \leq 1$ and $\varepsilon_{\text{opt}} B \geq \sum_{j=1}^{K} c_j$ hold. Substituting ε_{opt} into Equation 4.26, we get the required upper bound. □

That is, with a properly tuned value of ε, the budget–limited ε–first approach achieves a PAC upper bound of $O\left(B^{\frac{2}{3}}\right)$. This implies that the budget–limited ε–first approach satisfies Requirement 3 with high probability, since the $O\left(B^{\frac{2}{3}}\right)$ upper bound guarantees the optimal asymptotic convergence property within a PAC manner.

From the perspective of computational cost, recall that the uniform exploration policy has linear computational cost (i.e. $O(\varepsilon K B)$), since it sequentially pulls the arms. If T is the total number of pulls T within the exploration phase, from Equation 4.5 we get:

$$K \left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor \leq T \leq K \left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor + K \frac{\sum_{j=1}^{K} c_j}{c_{\min}}.$$

After the exploration phase, the budget–limited ε–first approach uses the density–ordered greedy algorithm once to estimate the best combination of pulls. Note this algorithm has a computational cost of $O(K \ln K)$ (see Section 2.5.2 for more details). This implies that the total computational cost within the exploitation phase is also $O(K \ln K)$. Given this, the total computational cost is $O\left(K \left\lfloor \frac{\varepsilon B}{\sum_{j=1}^{K} c_j} \right\rfloor + K \frac{\sum_{j=1}^{K} c_j}{c_{\min}} + K \ln K\right)$ In other words, the budget–limited ε–first approach satisfies Requirement 2, since it has low computational cost, compared to the size of the budget B and the number of arms K.

4.3 Summary

In this chapter, we developed a novel pulling algorithm, the budget–limited ϵ–first, for the budget–limited multi–armed bandit problem. In particular, this algorithm takes the first ε portion of the budget B to estimate the expected reward value of the arms (i.e. exploration), using the uniform pull policy. Based on these estimates, it approximates an unbounded knapsack problem in order to determine the best combination of arms that maximises the total expected reward, with respect to the residual budget $(1 - \varepsilon) B$ (i.e. exploitation). To approximate this unbounded knapsack, the algorithm uses a density–ordered greedy algorithm to approximate the best combination of arms. We showed that the budget–limited ϵ–first approach achieves linear regret bound with any value of ε (Theorem 4.1. This, however, is not efficient, and thus, does not satisfy Requirement

3 (i.e. efficient finite–time regret bound). In order to improve this result, we analysed the performance of the budget–limited ϵ–first approach from the PAC perspective. In more detail, we proved that within the PAC manner, the regret bound of the budget–limited ϵ–first approach with *any* exploration policy can be improved to be $2+\varepsilon B d_{\max}+ B\left(\sqrt{\frac{-\ln\frac{\beta}{2}}{n_{I^*}}}+\sqrt{\frac{-\ln\frac{\beta}{2}}{n_{I^+}}}\right)$ (Theorem 4.3). We refined this result in the case of uniform exploration policy (Corollary 4.8). In addition, we showed that the latter PAC bound can be further improved to be $O\left(B^{\frac{2}{3}}\right)$ if an optimal ε is chosen (Corollary 4.9).

Computation–wise, we demonstrated that the budget–limited ϵ–first approach typically has low computational cost. In particular, we showed that it has $O\left(\varepsilon KB + K\ln K\right)$ computational complexity. This implies that the budget–limited ϵ–first approach fully satisfies Requirement 2 (i.e. computational feasibility). In addition, we will demonstrate later in Chapter 7 that the budget–limited ϵ–first approach provides efficient performance in the problem of longterm information collection of WSNs. That is, it is efficient in terms of fulfilling Requirement 1 (i.e. efficient experimental performance quality).

However, the performance regret bound $O\left(B^{\frac{2}{3}}\right)$ of the budget–limited ε–first approach is only guaranteed with a certain probability, and thus, it might not hold for a number of cases. From this perspective, the budget–limited ε–first approach fails to satisfy Requirement 3. Given this, in the next chapters, we address this research requirement in terms of focusing on pulling algorithms with efficient theoretical regret bounds that guarantee the asymptotic optimal convergence. In particular, we provide two UCB–based pulling algorithms in Chapter 5, and two decreasing ε–greedy based algorithms in Chapter 6.

Chapter 5

Budget–Limited Upper Confidence Bound based Approaches

We now turn our attention to pulling algorithms that efficiently fulfil our Requirement 3, that is, they are designed to provide low theoretical regret bounds. Within this chapter, we focus on two upper confidence bound (UCB) based approaches, the knapsack based upper confidence bound exploration and exploitation (KUBE), and the fractional KUBE. To this end, we first introduce the algorithms in Section 5.1. We then provide logarithmic regret bounds for both algorithms in Section 5.2. In addition, we also show that these regret bounds are asymptotically optimal; that is, they only differ from the best possible bound with a constant factor.

5.1 The Algorithms

In this section, we thoroughly describe KUBE and its fractional counterpart. As mentioned in Chapter 1, the algorithms differ in the way they approximate the underlying unbounded knapsack problem at each time step. Given this, we first start with the discussion of KUBE, detailing how the algorithm is defined by combining the UCB based pulling policy with the density–ordered greedy algorithm (Section 5.1.1). Following this, we turn to describe the fractional KUBE, focusing on how it is different from KUBE (Section 5.1.2).

5.1.1 KUBE

To begin, consider the KUBE algorithm depicted in Algorithm 5.1. At each time step t, it first checks whether arm pulling is feasible (steps $3-4$). If the arm pulling is still feasible, KUBE first pulls each arm once in the initial phase (steps $6-7$). Following this, at each time step $t > K$, it estimates the best combination of arms according to their upper confidence bound using the density–ordered greedy approximation method applied to the following problem:

$$\max \sum_{i=1}^{K} m_{i,t} \left(\hat{\mu}_{i,n_{i,t}} + \sqrt{\frac{2 \ln t}{n_{i,t}}} \right) \quad \text{s.t.} \quad \sum_{i=1}^{K} m_{i,t} c_i \leq B_t, \; \forall i, t : m_{i,t} \text{ integer.} \tag{5.1}$$

In the above expression, $n_{i,t}$ is the number of pulls of arm i until time step t, $\sqrt{\frac{2 \ln t}{n_{i,t}}}$ is the size of the upper confidence interval, and $\hat{\mu}_{i,n_{i,t}}$ is the current estimate of arm i's expected reward, calculated as the average reward received so far from pulling arm i. More specifically, let $i(\tau)$ and $r(\tau)$ denote the arm chosen to be pulled and the received reward value at time step τ, respectively. Given this, $\hat{\mu}_{i,n_{i,t}}$ can be calculated as:

$$\hat{\mu}_{i,n_{i,t}} = \frac{1}{n_{i,t}} \sum_{\tau=1}^{t} \mathbf{I}_{\{i(\tau)=i\}} r(\tau), \tag{5.2}$$

where $\mathbf{I}_{\{i(\tau)=i\}}$ is the indicator function of the event $\{i(\tau) = i\}$ (i.e. the arm is pulled at time step τ is i). The goal, then, is to find integers $\{m_{i,t}\}_{i \in K}$ such that Equation 5.1 is maximised, with respect to the residual budget limit B_t (for the sake of simplicity, from here on, we drop the subscript $i \in K$ on this set). Since this problem is NP–hard, we use the density–ordered greedy method to find a near–optimal combination of arms (step 9). Note that the upper confidence bound on arm i's expected reward density is:

$$\frac{\hat{\mu}_{i,n_{i,t}}}{c_i} + \frac{\sqrt{\frac{2 \ln t}{n_{i,t}}}}{c_i}.$$

Let $M^*(B_t) = \{m^*_{i,t}\}$ be this method's solution to the problem in Equation 5.1, giving us the desired combination of arms, where $m^*_{i,t}$ is the number of arm i's pulls in the combination. Using $\{m^*_{i,t}\}$, KUBE *randomly* chooses the next arm to pull, $i(t)$, by selecting arm i with probability (step 10):

$$P(i(t) = i) = \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}}.$$

After the pull, it then updates the estimated upper bound of the chosen arm, and the residual budget limit B_t (steps $12-13$).

Algorithm 5.1 The KUBE Algorithm

$t = 1$; $B_t = B$;
while pulling is feasible **do**
  if $B_t < \min_i c_i$ **then**
    STOP! {pulling is not feasible}
  end if
  if $t \leq K$ **then**
    Initial phase: play arm $i(t) = t$;
  else
    use density–ordered greedy to calculate $M^*(B_t) = \{m^*_{i,t}\}$, the solution of Equation 5.1;
    randomly pull $i(t)$ with $P(i(t) = i) = \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}}$;
  end if
  update the estimated upper bound of arm $i(t)$;
  $B_{t+1} = B_t - c_{i(t)}$; $t = t + 1$;
end while

Algorithm 5.2 The Fractional KUBE Algorithm

$t = 1$; $B_t = B$;
while pulling is feasible **do**
  if $B_t < \min_i c_i$ **then**
    STOP! {pulling is not feasible}
  end if
  if $t \leq K$ **then**
    Initial phase: play arm $i(t) = t$;
  else
    pull arm $i(t) = \hat{I}(t)$, where $\hat{I}(t)$ is defined in Equation 5.4;
  end if
  update the estimated upper bound of arm $i(t)$;
  $B_{t+1} = B_t - c_{i(t)}$; $t = t + 1$;
end while

The intuition behind KUBE is the following: By repeatedly drawing the next arm to pull from a distribution formed by the current estimated approximate best combination, the expected reward of KUBE equals the average reward for following the optimal solution to the corresponding unbounded knapsack problem, given the current reward estimates. If the true values of the arms were known, then this would imply that the average performance of KUBE efficiently converges to the optimal solution of the unbounded knapsack problem reduced from the budget–limited MAB model. It is easy to show that the optimal solution of this knapsack model forms the theoretical optimal policy of the budget–limited MAB. In particular, if the mean reward value of each arm is known, then the budget–limited problem can be reduced to the unbounded knapsack problem, and thus, the optimal solution of the knapsack problem is the optimal solution of the budget–limited MAB as well. In addition, by combining the upper confidence bound with the estimated mean values of the arms, we guarantee that an arm that is not yet sampled many times may be pulled more frequently, since its upper confidence interval is large. Thus, we explore and exploit at the same time (for more details, see (Agrawal, 1995b; Audibert *et al.*, 2009; Auer *et al.*, 2002; Auer and Ortner, 2010)). By using the density–ordered greedy method at each time step, KUBE achieves an efficiently low regret bound by converging to the theoretical optimal solution, as detailed in the next section.

5.1.2 Fractional KUBE

We now turn to the fractional version of KUBE, which follows the underlying concept of KUBE. It also approximates the underlying unbounded knapsack problem at each time step t in order to determine the frequency of arms within the estimated best combination of arms. However, it differs from KUBE by using the fractional relaxation (see Section 2.5.2) to approximate the unbounded knapsack in Step 9 of Algorithm 5.1. Crucially, fractional KUBE uses the fractional relaxation based algorithm to solve the following fractional unbounded knapsack problem at each t:

$$\max \sum_{i=1}^{K} m_{i,t} \left(\hat{\mu}_{i,n_{i,t}} + \sqrt{\frac{2 \ln t}{n_{i,t}}} \right) \quad \text{s.t.} \quad \sum_{i=1}^{K} m_{i,t} c_i \leq B_t. \tag{5.3}$$

Recall that within KUBE, the frequency of arms within the approximated solution of the unbounded knapsack forms a probability distribution from which the agent randomly pulls the next arm. Now, since the fractional relaxation based algorithm solely chooses the arm (i.e. item type) with the highest estimated confidence bound–cost ratio (i.e. item density), fractional KUBE does not need to randomly choose an arm. Instead, at each time step t, it pulls the arm that maximises $\left(\frac{\hat{\mu}_{i,n_{i,t}}}{c_i} + \frac{\sqrt{\frac{2 \ln t}{n_{i,t}}}}{c_i} \right)$. That is, at each

time step i, fractional KUBE pulls arm $\hat{I}(t)$, such that:

$$\hat{I}(t) = \arg\max_{j} \left\{ \frac{\hat{\mu}_{j,t}}{c_j} + \frac{\sqrt{\frac{2\ln t}{n_{j,t}}}}{c_j} \right\}. \tag{5.4}$$

The fractional KUBE is depicted in Algorithm 5.2. Note that fractional KUBE can also be seen as the budget–limited version of UCB (see Section 2.2 for more details of UCB).

In the next section, we show that both KUBE and its fractional counterpart achieve asymptotically optimal regret bounds. That is, we first show that both algorithms achieve logarithmic regret bounds. Then we prove that these bounds only differ from the best possible one by a constant factor.

5.2 Performance Analysis

In this section, we first focus on the performance analysis of KUBE. To this end, we introduce some further notation. Let T denote the number of pulls of KUBE. In addition, let $N_j(T)$ denote the number of times KUBE pulls arm j up to time step T. In what follows, we first devise an upper bound for $N_j(T)$ for all $j \neq I^*$. That is, we estimate the number of times we pull arm $j \neq I^*$, instead of I^*. Based on this result, we estimate the average number of pulls of KUBE (i.e. $\mathbf{E}[T]$). This bound guarantees that KUBE always pulls "enough" arms so that the difference between the number of pulls in the theoretical optimal solution and that of KUBE is small, compared to the size of the budget. By using the estimated value of $\mathbf{E}[T]$, we then show that KUBE achieves a $O(\ln(B))$ worst case regret on average. We now state the following:

Lemma 5.1. *Suppose that* KUBE *pulls the arms T times. If $j \neq I^*$, then:*

$$\mathbf{E}[N_j(T)|T] \leq \left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}} \right)^2 \right) \ln(T) + \frac{\pi^2}{3} + 1.$$

That is, the number of times KUBE pulls an arm $j \neq I^*$ is at most $O(\ln(T))$. To prove this lemma, let us first refresh some of the terms that are used: $i(t)$ is the arm pulled by KUBE at time t; when refering to a combination of arms $\{m_{j,t}\}$, $m_{j,t}$ is the number times arm j is involved within this combination at time t; $M^*(B_t) = \{m^*_{i,t}\}$ is the density–ordered greedy approximate solution to unbounded knapsack problem in Equation 5.1, where $m^*_{i,t}$ is the number of arm i's pulls in this combination; and $I^* = \arg\max_i \frac{\mu_i}{c_i}$ is the arm with the highest true mean value density. In addition, $\hat{I}(t) = \arg\max_j \left\{ \frac{\hat{\mu}_{j,n_{j,t}}}{c_j} + \frac{\sqrt{\frac{2\ln t}{n_{j,t}}}}{c_j} \right\}$ is the arm with the highest estimated density confidence bound at time step t. In order to prove Lemma 5.1, we rely on the following lemmas:

Lemma 5.2. *Suppose that the total number of pulls* KUBE *makes of the arms is T, and that at each time step t, the residual budget is B_t (note that here $B_1 = B$). For any $0 < t \leq T$, we have:*

$$\frac{c_{\min}}{B_t} \leq \frac{1}{T-t+1}.$$

Lemma 5.3. *Suppose that the total number of pulls* KUBE *makes of the arms is T. For any $0 < t \leq T$, we have:*

$$P\left(i\left(t\right)=j|\,T\right) \leq P\left(\hat{I}\left(t\right)=j|T\right)+\left(\frac{c_{\max}}{c_{\min}}\right)^{2}\frac{1}{T-t+1}.$$

Proof of Lemma 5.2. At the beginning of time step t, the residual budget is B_t. Since the total number of pulls is T, with respect to B_t, KUBE can still make $T-t+1$ pulls (including the pull at time step t). This indicates that:

$$B_t \geq c_{i(t)} + c_{i(t+1)} + \cdots + c_{i(T)} \geq (T-t+1)\, c_{\min}.$$

which directly implies the inequality in Lemma 5.2. □

Proof of Lemma 5.3. We assume that the value of T is given. For the slight abuse of notation, we drop the conditional of T notation to simplify the proof (i.e. all the probabilities are considered to be conditional to T), and we will explicitly denote it when necessary. First, we consider a particular value of B_t. Thus, we have:

$$P\left(i\left(t\right)=j|B_t\right)=\sum_{\{m_{i,t}\}}P\left(i\left(t\right)=j|M^{*}\left(B_t\right)=\{m_{i,t}\}\right)P\left(M^{*}\left(B_t\right)=\{m_{i,t}\}\right). \tag{5.5}$$

Recall that the density–ordered greedy approach first repeatedly adds arm $\hat{I}\left(t\right)$ to combination $\{m_{i,t}\}$ until it is not feasible. It is easy to show that after adding arm $\hat{I}\left(t\right)$ as many times as possible (i.e. $m_{\hat{I}(t),t}$ times) to the combination, the residual budget is at most $c_{\hat{I}(t)}$ (or otherwise we could still add arm $\hat{I}\left(t\right)$ one more time). Therefore:

$$\sum_{i\neq\hat{I}(t)} m_{i,t} \leq \frac{c_{\hat{I}(t)}}{c_{\min}}. \tag{5.6}$$

That is, the total count of arm pulls other than $\hat{I}\left(t\right)$ in the combination is at most $\frac{c_{\hat{I}(t)}}{c_{\min}}$. This inequality comes from the fact that we can construct a combination with the greatest number of arm pulls by only adding the arm with the smallest cost. Similarly, we have:

$$\sum_{k=1}^{K} m_{k,t} \geq \frac{B_t}{c_{\max}}, \tag{5.7}$$

because we can construct a combination with the smallest number of arm pulls by only adding the arm with the greatest cost. Combining Equations 5.6 and 5.7 gives:

$$\frac{\sum_{i\neq\hat{I}(t)} m_{i,t}}{\sum_{k=1}^{K} m_{k,t}} \leq \frac{\frac{c_{\hat{I}(t)}}{c_{\min}}}{\frac{B_t}{c_{\max}}} \leq \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{c_{\min}}{B_t}. \tag{5.8}$$

The last inequality is obtained from the fact that $c_{\hat{I}(t)} \leq c_{\max}$. Now, recall that KUBE chooses arm j to pull with probability $\frac{m_{j,t}}{\sum_{k=1}^{K} m_{k,t}}$. This implies that:

$$\begin{aligned} &P(i\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\,) \\ &\qquad = P\left(i\,(t) = j, \hat{I}\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\right) \\ &\qquad + P\left(i\,(t) = j, \hat{I}\,(t) \neq j | M^*\,(B_t) = \{m_{i,t}\}\right). \end{aligned}$$

This can be bounded by:

$$\begin{aligned} &P(i\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\,) \\ &\qquad \leq \frac{m_{\hat{I}(t),t}}{\sum_{k=1}^{K} m_{k,t}} P\left(\hat{I}\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\right) \\ &\qquad + \frac{\sum_{i\neq\hat{I}(t)} m_{i,t}}{\sum_{k=1}^{K} m_{k,t}} P\left(\hat{I}\,(t) \neq j | M^*\,(B_t) = \{m_{i,t}\}\right). \end{aligned} \tag{5.9}$$

The right hand side can be further bounded as follows:

$$\begin{aligned} &P(i\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\,) \\ &\qquad \leq P\left(\hat{I}\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\right) + \frac{\sum_{i\neq\hat{I}(t)} m_{i,t}}{\sum_{k=1}^{K} m_{k,t}} \\ &\qquad \leq P\left(\hat{I}\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\right) + \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{c_{\min}}{B_t}. \end{aligned} \tag{5.10}$$

The last inequality is obtained from Equation 5.8. Substituting Equation 5.10 into Equation 5.5 gives:

$$\begin{aligned} P\,(i\,(t) = j | B_t) &\leq \sum_{\{m_{i,t}\}} \left(P\left(\hat{I}\,(t) = j | M^*\,(B_t) = \{m_{i,t}\}\right) + \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{c_{\min}}{B_t}\right) P\,(M^*\,(B_t) = \{m_{i,t}\}) \\ &\leq P\left(\hat{I}\,(t) = j | B_t\right) + \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{c_{\min}}{B_t} \\ &\leq P\left(\hat{I}\,(t) = j | B_t\right) + \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{1}{T-t+1}. \end{aligned} \tag{5.11}$$

The last inequality is obtained from Lemma 5.2. Now we study the general case, where B_t is not fixed. By summing up Equation 5.11 over all possible value of B_t, we have:

$$\begin{aligned} P\left(i\left(t\right)=j|T\right) &= \sum_{B_t} P\left(i\left(t\right)=j|T,B_t\right) P\left(B_t|T\right) \\ &\leq \sum_{B_t} \left(P\left(\hat{I}\left(t\right)=j|T,B_t\right) + \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{1}{T-t+1} \right) P\left(B_t|T\right) \\ &\leq P\left(\hat{I}\left(t\right)=j|T\right) + \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{1}{T-t+1}. \end{aligned} \tag{5.12}$$

which concludes the proof. □

Based on Lemmas 5.2 and 5.3, Lemma 5.1 can be proved as follows:

Proof of Lemma 5.1. We assume that the value of T is already given. Again, for the slight abuse of notation, we drop the conditional of T notation to simplify the proof, and we will explicitly denote it when necessary. In this case, the proof of the theorem for that particular value of T is along the same lines as that of Theorem 1 of Auer *et al.* (2002). In particular, recall that $N_j\left(T\right)$ denotes the expectation of number of times KUBE pulls an arm $j \neq I^*$ until time step T. Given this, we have the following:

$$\begin{aligned} \mathbf{E}\left[N_j\left(T\right)\right] &= 1 + \sum_{t=K+1}^{T} P\left(i\left(t\right)=j\right) \\ &\leq 1 + \sum_{t=K+1}^{T} P\left(\hat{I}\left(t\right)=j\right) + \sum_{t=K+1}^{T} \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{1}{T-t+1} \\ &\leq l + \sum_{t=K+1}^{T} P\left(\hat{I}\left(t\right)=j, N_j\left(t\right)\geq l\right) + \sum_{t=K+1}^{T} \left(\frac{c_{\max}}{c_{\min}}\right)^2 \frac{1}{T-t+1} \end{aligned} \tag{5.13}$$

for any $l \geq 1$. Now, let $b_{t,s} = \sqrt{\frac{2\ln t}{s}}$. Considering the second term on the right hand side of Equation 5.13, we have:

$$\begin{aligned} \sum_{t=K+1}^{T} P\left(\hat{I}\left(t\right)=j, N_j\left(t\right)\geq l\right) &= \sum_{t=K+1}^{T} P\left(\frac{\hat{\mu}_{I^*,N_{I^*}(t)}}{c_{I^*}} + \frac{b_{t,N_{I^*}(t)}}{c_{I^*}} \leq \frac{\hat{\mu}_{j,N_j(t)}}{c_j} + \frac{b_{t,N_j(t)}}{c_j}, N_j\left(t\right)\geq l\right) \\ &\leq \sum_{t=K+1}^{T} P\left(\min_{1\leq s\leq t}\left\{\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}}\right\} \leq \max_{l\leq s_j\leq t}\left\{\frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}\right\}\right) \\ &\leq \sum_{t=1}^{T}\sum_{s=1}^{t}\sum_{s_j=1}^{t} P\left(\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}} \leq \frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}\right). \end{aligned} \tag{5.14}$$

If it is true that $\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}} \leq \frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}$, then at least one of the following three statements must also hold:

$$\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}} \leq \frac{\mu_{I^*}}{c_{I^*}}, \tag{5.15}$$

$$\frac{\mu_j}{c_j} \leq \frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}, \tag{5.16}$$

$$\frac{\mu_{I^*}}{c_{I^*}} \leq \frac{\mu_j}{c_j} + \frac{2b_{t,s_j}}{c_j}. \tag{5.17}$$

That is, we get:

$$\begin{aligned} P\left(\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}} \leq \frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}\right) \leq & P\left(\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}} \leq \frac{\mu_{I^*}}{c_{I^*}}\right) + \\ & + P\left(\frac{\mu_j}{c_j} \leq \frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}\right) + P\left(\frac{\mu_{I^*}}{c_{I^*}} \leq \frac{\mu_j}{c_j} + \frac{2b_{t,s_j}}{c_j}\right). \end{aligned} \tag{5.18}$$

Applying the Chernoff–Hoeffding inequalities to the first two terms on the right hand side of Equation 5.18 gives:

$$P\left(\frac{\hat{\mu}_{I^*,s}}{c_{I^*}} + \frac{b_{t,s}}{c_{I^*}} \leq \frac{\mu_{I^*}}{c_{I^*}}\right) = P\left(\hat{\mu}_{I^*,s} + b_{t,s} \leq \mu_{I^*}\right) \leq \exp\left\{-2b_{t,s}^2 s\right\} = \exp\left\{-4\ln t\right\} = t^{-4} \tag{5.19}$$

$$P\left(\frac{\mu_j}{c_j} \leq \frac{\hat{\mu}_{j,s_j}}{c_j} + \frac{b_{t,s_j}}{c_j}\right) = P\left(\mu_j \leq \hat{\mu}_{j,s_j} + b_{t,s_j}\right) \leq \exp\left\{-2b_{t,s_j}^2 s_j\right\} = \exp\left\{-4\ln t\right\} = t^{-4}. \tag{5.20}$$

On the other hand, for $l \geq \frac{8\ln T}{d_{\min}^2}$, Equation 5.17 is false, since:

$$\begin{aligned} \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} - \frac{2b_{t,s_j}}{c_j} &\geq \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} - 2b_{t,s_j} \\ &\geq \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} - 2\sqrt{\frac{2\ln t}{l}} \\ &\geq \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} - 2\sqrt{\frac{2\ln t}{\frac{8\ln T}{d_{\min}^2}}} \\ &\geq \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} - d_{\min} \\ &\geq \frac{\mu_{I^*}}{c_{I^*}} - \frac{\mu_j}{c_j} - d_j = 0. \end{aligned} \tag{5.21}$$

Here note that $c_j \geq 1$, $s_j \geq l \geq \frac{8\ln T}{d_{\min}^2}$, and $t \leq T$. If $l \geq \frac{8\ln T}{d_{\min}^2}$, then $P\left(\frac{\mu_{I^*}}{c_{I^*}} \leq \frac{\mu_j}{c_j} + \frac{2b_{t,s_j}}{c_j}\right) = 0$. Substituting this and Equations 5.18, 5.19 and 5.20 into Equation 5.14 gives:

$$\sum_{t=K+1}^{T} P\left(\hat{I}\left(t\right) = j, N_j\left(t\right) \geq l\right) \leq \sum_{t=1}^{T}\sum_{s=1}^{t}\sum_{s_j=1}^{t} 2t^{-4} \leq \frac{\pi^2}{3}, \tag{5.22}$$

for any $l \geq \left\lceil \frac{8\ln T}{d_{\min}^2} \right\rceil$. Note that the last inequality is obtained from the Riemann Zeta Function for value of 2 (i.e. $\sum_{t=1}^{\infty} t^{-2} = \frac{\pi^2}{6}$) (Ivic, 1985).

Now, consider the third term on the right hand side of Equation 5.13. By using Lemma 5.2, we get:

$$\sum_{t=1}^{T}\left(\frac{c_{\max}}{c_{\min}}\right)^{2}\frac{1}{T-t+1}\leq\left(\frac{c_{\max}}{c_{\min}}\right)^{2}\ln\left(T\right). \tag{5.23}$$

We now combine Equations 5.22 and 5.23 together, and we set $l=\frac{8\ln T}{d_{\min}^{2}}+1$, which gives:

$$\mathbf{E}\left[N_{j}\left(T\right)\right]\leq\frac{8\ln T}{d_{\min}^{2}}+1+\frac{\pi^{2}}{3}+\left(\frac{c_{\max}}{c_{\min}}\right)^{2}\ln\left(T\right)$$

for any given value of T, which concludes the proof. □

From Lemma 5.1, we can show the following:

Lemma 5.4. *Suppose that the total budget size is B. If T denotes the total number of pulls of* KUBE *then we have:*

$$\mathbf{E}\left[T\right]\geq\frac{B}{c_{I^{*}}}-\left(\frac{8}{d_{\min}^{2}}+\left(\frac{c_{\max}}{c_{\min}}\right)^{2}\right)\sum_{\delta_{j}>0}\frac{\delta_{j}}{c_{I^{*}}}\ln\left(\frac{B}{c_{\min}}\right)-\sum_{\delta_{j}>0}\frac{\delta_{j}}{c_{I^{*}}}\left(\frac{\pi^{2}}{3}+1\right)-1$$

where $\mathbf{E}\left[T\right]$ is the expected number of pulls using KUBE*.*

That is, the difference between $\frac{B}{c_{I^{*}}}$ and the number of pulls of KUBE is at most $O\left(\ln\left(\frac{B}{c_{\min}}\right)\right)$.

Proof of Lemma 5.4. Since KUBE pulls arms until none are feasible, by definition:

$$P\left(\sum_{t=1}^{T}c_{i(t)}\leq B-c_{\min}\right)=1.$$

Taking the expectation of $\sum_{t=1}^{T}c_{i(t)}$ over T and $\{m_{j,t}\}$ (i.e. the set of $i\left(t\right)$) gives:

$$\begin{aligned}B-c_{\min}&\leq\mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T}c_{i(t)}\right]=\mathbf{E}_{T}\left[\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[c_{i(t)}\right]\right]\\&\leq\mathbf{E}_{T}\left[\sum_{t=1}^{T}\sum_{j=1}^{K}c_{j}P\left(i\left(t\right)=j|T\right)\right]\\&\leq\mathbf{E}_{T}\left[\sum_{t=1}^{T}\left(c_{I^{*}}+\sum_{\delta_{j}>0}\delta_{j}P\left(i\left(t\right)=j|T\right)\right)\right]\end{aligned}$$

$$\leq \mathbf{E}_T[T]\, c_{I^*} + \mathbf{E}_T\left[\sum_{\delta_j>0} \delta_j \left(\sum_{t=1}^{T} P\left(i\left(t\right)=j|T\right)\right)\right]$$

$$\leq \mathbf{E}_T[T]\, c_{I^*} + \mathbf{E}_T\left[\sum_{\delta_j>0} \delta_j \left(\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\ln\left(T\right) + \frac{\pi^2}{3} + 1\right)\right] \tag{5.24}$$

$$\leq \mathbf{E}_T[T]\, c_{I^*} + \sum_{\delta_j>0} \delta_j \left(\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\ln\left(\frac{B}{c_{\min}}\right) + \frac{\pi^2}{3} + 1\right). \tag{5.25}$$

Equation 5.24 is obtained from Lemma 5.1, while Equation 5.25 comes from the fact that $T \leq \frac{B}{c_{\min}}$ with probability 1. In addition, the third inequality is obtained from the fact that δ_j can be smaller than 0 for some j, and thus, we can further upper bound by only summing up $\delta_j P\left(i\left(t\right)=j|T\right)$ over arms that have $\delta_j > 0$. Now, by dividing both sides with c_{I^*}, we obtain:

$$\frac{B}{c_{I^*}} - \frac{c_{\min}}{c_{I^*}} - \sum_{\delta_j>0} \frac{\delta_j}{c_{I^*}} \left(\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\ln\left(\frac{B}{c_{\min}}\right) + \frac{\pi^2}{3} + 1\right) \leq \mathbf{E}_T[T].$$

By using the fact that $\frac{c_{\min}}{c_{I^*}} \leq 1$, we obtain the stated formula. □

Note that if we relax the budget–limited MAB problem so that the number of pulls can be fractional, then it is easy to show that the optimal pulling policy of this relaxed model is to repeatedly pull arm I^* only. In this case, $\frac{B}{c_{I^*}}$ is the number of pulls of this optimal policy. Lemma 5.4 indicates that the number of pulls that KUBE produces does not significantly differ from that of the optimal policy of the fractional budget–limited MAB (i.e. the difference is a logarithmic function of the number of pulls). We can now derive the regret bound of KUBE from Lemma 5.4 as follows:

Theorem 5.5. *For any budget size $B > 0$, the performance regret of* KUBE *is at most:*

$$\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\left(\sum_{\Delta_j>0} \Delta_j + \sum_{\delta_j>0} \frac{\delta_j}{c_{I^*}}\right)\ln\left(\frac{B}{c_{\min}}\right) + \left(\sum_{\Delta_j>0} \Delta_j + \sum_{\delta_j>0} \frac{\delta_j}{c_{I^*}}\right)\left(\frac{\pi^2}{3} + 1\right) + 1.$$

Note that since for each $j \neq I^*$, at least one between δ_j and Δ_j has to be positive (see Chapter 3 for more details), we can easily show that $\left(\sum_{\Delta_j>0} \Delta_j + \sum_{\delta_j>0} \frac{\delta_j}{c_{I^*}}\right) > 0$. That is, the performance regret of KUBE (i.e. $R^B\left(\text{KUBE}\right)$) is upper–bounded by $O\left(\ln\left(\frac{B}{c_{\min}}\right)\right)$.

Proof of Theorem 5.5. Recall that $\mathbf{E}\left[G^B\left(A^*\right)\right]$ denotes the expected performance of the theoretical optimal policy. It is obvious that $\mathbf{E}\left[G^B\left(A^*\right)\right] \leq \frac{B\mu_{I^*}}{c_{I^*}}$, since the latter is the

optimal solution of the fractional budget–limited MAB problem. This indicates that:

$$\begin{aligned}
R^B(\text{KUBE}) &= \mathbf{E}\left[G^B(A^*)\right] - \mathbf{E}\left[G^B(\text{KUBE})\right] \\
&\leq \frac{B\mu_{I^*}}{c_{I^*}} - \mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T}\mu_{i(t)}\right] \\
&\leq \frac{B\mu_{I^*}}{c_{I^*}} - \mathbf{E}_T\left[\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right] \\
&\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right] \\
&\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T}\sum_{j}^{K}\mu_j P\left(i\left(t\right)=j|T\right)\right] \\
&\leq \mathbf{E}_T\left[\left(\frac{B}{c_{I^*}} - T\right)\mu_{I^*} + \sum_{t=1}^{T}\left(\mu_{I^*} - \sum_{j}^{K}\mu_j P\left(i\left(t\right)=j|T\right)\right)\right] \\
&\leq \mathbf{E}_T\left[\frac{B}{c_{I^*}} - T\right]\mu_{I^*} + \mathbf{E}_T\left[\sum_{t=1}^{T}\sum_{\Delta_j>0}\Delta_j P\left(i\left(t\right)=j|T\right)\right] \\
&\leq \mathbf{E}_T\left[\frac{B}{c_{I^*}} - T\right]\mu_{I^*} + \mathbf{E}_T\left[\sum_{\Delta_j>0}\Delta_j \mathbf{E}\left[N_j\left(T\right)|T\right]\right]. \qquad (5.26)
\end{aligned}$$

Note that since Δ_j can be smaller than 0 for some arm j, we can further upper bound $R^B(\text{KUBE})$ by only summing up $\Delta_j \mathbf{E}\left[N_j\left(T\right)|T\right]$ over arms with $\Delta_j > 0$ (see the last two inequalities). Applying Lemma 5.4 to the first term and Lemma 5.1 to the second term on the right hand side of Equation 5.26 gives:

$$\begin{aligned}
R^B(\text{KUBE}) \leq &\left[\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\ln\left(\frac{B}{c_{\min}}\right) + \sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\left(\frac{\pi^2}{3}+1\right)+1\right]\mu_{I^*}+ \\
&+ \mathbf{E}_T\left[\sum_{\Delta_j>0}\Delta_j\left(\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\ln\left(T\right) + \frac{\pi^2}{3}+1\right)\right] \\
&\leq \left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\ln\left(\frac{B}{c_{\min}}\right) + \sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\left(\frac{\pi^2}{3}+1\right)+1+ \\
&+ \sum_{\Delta_j>0}\Delta_j\left(\left(\frac{8}{d_{\min}^2} + \left(\frac{c_{\max}}{c_{\min}}\right)^2\right)\ln\left(\frac{B}{c_{\min}}\right) + \frac{\pi^2}{3}+1\right)
\end{aligned}$$

which concludes the proof. Note that the last equation is obtained from the facts that $\mu_{I^*} \leq 1$ and $T \leq \frac{B}{c_{\min}}$ with probability 1. □

In a similar vein, we can show that the regret of fractional KUBE is bounded as follows:

Theorem 5.6. *For any budget size $B > 0$, the performance regret of fractional* KUBE *is at most*

$$\frac{8}{d_{\min}^2}\left(\sum_{\Delta_j>0}\Delta_j+\sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\right)\ln\left(\frac{B}{c_{\min}}\right)+\left(\sum_{\Delta_j>0}\Delta_j+\sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\right)\left(\frac{\pi^2}{3}+1\right)+1\,.$$

Proof. We follow the concept that is similar to the proof of Theorem 5.5. Given this, we only highlight the steps that are different from the previous proofs. For the sake of simplicity, we use the notations previously introduced for the performance analysis of KUBE. In particular, let T denote the random variable that represents the number of pulls that fractional KUBE uses. Let $N_j(T)$ denote the number of times that the corresponding pulling algorithm pulls arm j up to time step T. Similar to Lemma 5.1, we first show that within the fractional KUBE algorithm, we have:

$$\mathbf{E}\left[N_j(T)\,|T\right]\leq\frac{8}{d_{\min}^2}\ln(T)+\frac{\pi^2}{3}+1. \tag{5.27}$$

In so doing, note that

$$\mathbf{E}\left[N_j(T)\,|T\right]=1+\sum_{t=K+1}^{T}P\left(i(t)=j|T\right)\leq l+\sum_{t=K+1}^{T}P\left(i(t)=j,N_j(t)\geq l|T\right) \tag{5.28}$$

for any $l\geq 1$. Now, using similar techniques from the proof of Lemma 5.1, we can easily show that

$$\sum_{t=K+1}^{T}P\left(i(t)=j,N_j(t)\geq l|T\right)\leq\sum_{t=1}^{T}\sum_{s=1}^{t}\sum_{s_j=1}^{t}2t^{-4}\leq\frac{\pi^2}{3},$$

for any $l\geq\left\lceil\frac{8\ln T}{d_{\min}^2}\right\rceil$. By substituting this into Equation 5.28, we obtain Equation 5.27. Next, we show that

$$\mathbf{E}[T]\geq\frac{B}{c_{I^*}}-\frac{8}{d_{\min}^2}\sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\ln\left(\frac{B}{c_{\min}}\right)-\sum_{\delta_j>0}\frac{\delta_j}{c_{I^*}}\left(\frac{\pi^2}{3}+1\right)-1. \tag{5.29}$$

This can be derived from Equation 5.27 by using techniques similar to the proof of Lemma 5.4. This implies that

$$\begin{aligned}
R^B(\text{KUBE}) &= \mathbf{E}\left[G^B(A^*)\right]-\mathbf{E}\left[G^B(\text{KUBE})\right]\\
&\leq\frac{B\mu_{I^*}}{c_{I^*}}-\mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T}\mu_{i(t)}\right]\\
&\leq\frac{B\mu_{I^*}}{c_{I^*}}-\mathbf{E}_T\left[\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right]\\
&\leq\mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}}-\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right]
\end{aligned}$$

$$\leq \mathbf{E}_T \left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T} \sum_{j}^{K} \mu_j P\left(i\left(t\right) = j|T\right) \right]$$

$$\leq \mathbf{E}_T \left[\left(\frac{B}{c_{I^*}} - T \right) \mu_{I^*} + \sum_{t=1}^{T} \left(\mu_{I^*} - \sum_{j}^{K} \mu_j P\left(i\left(t\right) = j|T\right) \right) \right]$$

$$\leq \mathbf{E}_T \left[\frac{B}{c_{I^*}} - T \right] \mu_{I^*} + \mathbf{E}_T \left[\sum_{t=1}^{T} \sum_{\Delta_j > 0} \Delta_j P\left(i\left(t\right) = j|T\right) \right]$$

$$\leq \mathbf{E}_T \left[\frac{B}{c_{I^*}} - T \right] \mu_{I^*} + \mathbf{E}_T \left[\sum_{\Delta_j > 0} \Delta_j \mathbf{E}\left[N_j\left(T\right)|T\right] \right]. \tag{5.30}$$

By substituting Equations 5.28 and 5.29 into this, we obtain

$$R^B\left(\text{KUBE}\right) \leq \frac{8}{d_{\min}^2} \sum_{\delta_j > 0} \frac{\delta_j}{c_{I^*}} \ln\left(\frac{B}{c_{\min}}\right) + \sum_{\delta_j > 0} \frac{\delta_j}{c_{I^*}} \left(\frac{\pi^2}{3} + 1\right) + 1 +$$
$$+ \sum_{\Delta_j > 0} \Delta_j \left(\frac{8}{d_{\min}^2} \ln\left(\frac{B}{c_{\min}}\right) + \frac{\pi^2}{3} + 1 \right)$$

which concludes the proof. □

Having established a regret bound for the two algorithms, we now move on to show that they produce optimal behaviour, in terms of minimising the regret.

Theorem 5.7. *For any arm pulling algorithm, there exists a constant $C \geq 0$, and a particular instance of the budget–limited MAB problem, such that the regret of that algorithm within that particular problem is at least $C \ln \frac{B}{c_{\min}}$.*

Proof. By setting all of the arms' pulling costs equal to $c \geq 0$, any standard MAB problem can be reduced to a budget–limited MAB. This implies that the number of pulls within this MAB is guaranteed to be $\frac{B}{c} = T$ (i.e. T is deterministic). According to Lai and Robbins (1985), the best possible regret that an arm pulling algorithm can achieve within the domain of standard MABs is $C \ln\left(T\right)$. Therefore, if there is an algorithm within the domain of budget–limited that provides better regret than $C \ln\left(\frac{B}{c_{\min}}\right) = C \ln T$, then it also provides better regret bounds for standard MABs. □

Now, since the performance regret of both algorithms is $O\left(\ln\left(\frac{B}{c_{\min}}\right)\right)$, Theorem 5.7 indicates that their performance is asymptotically optimal (i.e. their performance differs from that of the optimal policy by a constant factor). That is, it is easy to show that both KUBE and its fractional counterpart follow the concept of asymptotic optimal convergence, and thus, they fulfil Requirement 3.

Computation–wise, at each time step t, KUBE uses a density–ordered greedy algorithm to approximate the solution of the underlying unbounded knapsack problem. This indicates that at each time step, the computational cost of KUBE is $O(K \ln K)$ (see Section 2.5.2 for more details). Recall that T is random variable that represents the number of pulls of KUBE. It is easy to show that:

$$T \leq \frac{B}{c_{\min}}$$

with probability 1. Note the right hand side is the number of pulls when we repeatedly pull the arm with the lowest pulling cost. Thus, the number of pulls is always bounded by $\frac{B}{c_{\min}}$, since we can achieve the maximal number of pulls if we only choose to pull the arm with $c_{\min}$ pulling cost. This implies that the total computational cost of KUBE is $O\left(\frac{BK \ln K}{c_{\min}}\right)$, which is low, compared to the budget size B and number of arms K. Thus, KUBE satisfies Requirement 2.

By replacing the density–ordered greedy with the fractional relaxation based algorithm, fractional KUBE decreases the computational cost to $O(K)$ per time step. More precisely, at each time step, fractional KUBE calculates $\hat{I}(t)$, that is arm with the highest confidence bound density (see Equation 5.4 for more detail). This can be evaluated with $O(K)$ computational cost. That is, the total computational cost of fractional KUBE is $O(BK)$, which is lower than that of KUBE. This implies that while both algorithms satisfy Requirement 2 (i.e. low computational complexity), KUBE is outperformed by its fractional counterpart.

5.3 Summary

In this chapter, we focused on developing pulling algorithms that fulfil Requirement 3 (i.e. efficient finite–time regret bound). To this end, we proposed two algorithms, KUBE and fractional KUBE, that combine the UCB based pulling technique with unbounded knapsack approximation methods. In particular, KUBE uses the current estimates of the expected reward values to form an underlying unbounded knapsack problem at each time step t. To solve this knapsack problem, it relies on a density–ordered greedy approximation approach. Similarly, fractional KUBE also solves an unbounded knapsack problem at each time step. However, it uses a fractional relaxation technique to approach the optimal solution of this knapsack problem. We showed that these algorithms provide efficient theoretical regret bounds that follow the concept of asymptotic optimal convergence; that is, they both efficiently satisfy Requirement 3. In more detail, we first proved that KUBE has a $O(\ln B)$ regret bound (Theorem 5.5). In so doing, we provided an upper bound for the number of times we pull a sub–optimal arm i (i.e. the arm that differs from I^*) in Lemma 5.1. Using this result, we then provided a lower bound for the

value of T, the number of pulls within KUBE (Lemma 5.4). These lemmas provide a basis to prove Theorem 5.5, which guarantees a logarithmic upper bound for the regret of KUBE. In a similar vein, we also showed that fractional KUBE achieves a logarithmic upper bound (Theorem 5.6). Following this, we proved that the aforementioned upper bounds are asymptotically optimal; that is, they only differ from the best possible by a constant factor (Theorem 5.7).

From the perspective of computational complexity, we pointed out that while KUBE has a $O(B(K + \ln K))$ computational cost, its fractional counterpart achieves a reduced cost of $O(BK)$. That is, both algorithms have low computational cost, compared to the budget size B, and the number of arms K. This indicates that the algorithms fulfil Requirement 2 (i.e. computational feasibility).

Although both KUBE and its fractional counterpart outperform the budget–limited ε–first approach in terms of fulfilling Requirement 3, as we will show later in Chapter 7, these algorithms typically provide poor performance in the scenario of long–term information collection of WSNs (i.e. they are significantly outperformed by the budget–limited ε–first approach), and thus, fail to satisfy Requirement 1 (i.e. efficient experimental performance quality). Against this background, in the next chapter, we propose a trade–off between the budget–limited ε–first and UCB based approaches, that performs well from both a theoretical and an empirical aspect.

Chapter 6

Budget–Limited Decreasing Epsilon–Greedy based Approaches

So far, we have developed pulling algorithms that follow the concepts of ε–first and UCB in order to find a trade–off between exploration and exploitation within budget–limited MABs. However, the budget–limited ε–first approach cannot guarantee efficient theoretical regret bounds, and thus, it fails to fulfil Requirement 3 (i.e. efficient finite–time regret bound). In contrast, both KUBE and fractional KUBE achieve asymptotic optimal regret bounds, but as we will show later in Chapter 7, they are outperformed by the budget–limited ε–first approach in real–world settings.

Hence, we identify a need for a pulling algorithm that shows good performance from both theoretical and experimental perspectives. To this end, within this chapter, we propose two decreasing ε–greedy based algorithms: (i) the knapsack based decreasing ε–greedy (KDE); and (ii) the fractional KDE. In so doing, we first describe the algorithms in Section 6.1. This is followed, in Section 6.2, by the performance analysis of KDE and its fractional counterpart. More precisely, we provide theoretical bounds on the performance regret of the algorithms, and we study their computational cost.

6.1 The Algorithms

In this section, we focus on the description of KDE and its fractional counterpart. Similar to the case of the UCB based algorithm in the previous chapter, these algorithms differ from each other in the way they approximate the underlying unbounded knapsack problem at each time step. Given this, we first start with the discussion of KDE, detailing

how the algorithm is defined by combining the decreasing ε–greedy based pulling policy with the density–ordered greedy algorithm (Section 6.1.1). Following this, we turn to the fractional KDE, focusing on how it is different from KDE (Section 6.1.2).

6.1.1 KDE

Consider the KDE algorithm depicted in Algorithm 6.1. Here, similarly to the previous chapters, t also denotes the time step. Furthermore, let B_t denote the residual budget at time t. Note that at the start (i.e. $t = 1$), $B_1 = B$, where B is the total budget limit. At each subsequent time step, t, KDE first checks whether arm pulling is no longer feasible. Note that it is infeasible if and only if none of the arms can be pulled, with the remaining budget. Specifically, if $B_t < \min_j c_j$ (i.e. the residual budget is smaller than the lowest pulling cost), then KDE stops (steps $3 - 4$). If the arm pulling is still feasible, KDE then estimates the best combination of arms, denoted with $M^*(B_t)$, by using the aforementioned density–ordered greedy approximation method (step 6). This method provides an approximation of the best combination with greatest estimated total expected reward, that does not exceed the residual budget limit B_t at t. In particular, at each time step t, the algorithm solves the unbounded knapsack problem by using the density–ordered greedy approximation method as follows. Similar to the case of KUBE in Section 5.1.1, KDE solves the following knapsack problem at each time step t:

$$\max \sum_{i=1}^{K} m_{i,t} \hat{\mu}_{i,n_{i,t}} \quad \text{s.t.} \quad \sum_{i=1}^{K} m_{i,t} c_i \leq B_t, \; \forall i, t : m_{i,t} \text{ integer}. \tag{6.1}$$

where $n_{i,t}$ denotes the number of times up to t when KDE pulls arm i, and $\hat{\mu}_{i,n_{i,t}}$ is the estimated value of arm i's expected reward, which is calculated as the average reward received so far for pulling arm i. More precisely, this estimate can be calculated as described in Equation 5.2 (see Section 5.1.1 for more detail). At each time step, KDE aims to find integers $\{m_{i,t}\}$ such that Equation 6.1 is maximised, with respect to the residual budget limit B_t. Since this problem is NP–hard, by using the density–ordered greedy method, we can achieve a near–optimal combination of arms. Let $\{m^*_{i,t}\}$ be the solution of this method to the knapsack problem in Equation 6.1. Thus, $M^*(B_t) = \{m^*_{i,t}\}$ gives us the desired combination of arms, where $m^*_{i,t}$ denotes the number of arm i's pulls within the combination. Next, let

$$\varepsilon_t = \min\{1, \frac{\gamma}{t}\}, \tag{6.2}$$

Algorithm 6.1 The KDE Algorithm

$t = 1$; $B_t = B$; $\gamma > 0$;
while pulling is feasible **do**
  if $B_t < \min_i c_i$ **then**
    STOP! {pulling is not feasible}
  end if
  use density–ordered greedy to calculate $M^*(B_t) = \{m^*_{i,t}\}$ {approximated best combination of arms by estimated values};
  $\varepsilon_t = \min\{1, \gamma/t\}$;
  randomly pull $i(t)$ with $P(i(t) = i) = (1 - \varepsilon_t)\frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}} + \frac{\varepsilon_t}{K}$;
  $B_{t+1} = B_t - c_{i(t)}$; $t = t + 1$;
end while

Algorithm 6.2 The Fractional KDE Algorithm

$t = 1$; $B_t = B$; $\gamma > 0$;
while pulling is feasible **do**
  if $B_t < \min_i c_i$ **then**
    STOP! {pulling is not feasible}
  end if
  $\varepsilon_t = \min\{1, \gamma/t\}$;
  let $P(i(t) = I^+(t)) = (1 - \varepsilon_t)$ and $P(i(t) = j, j \neq I^+(t)) = \frac{\varepsilon_t}{K}$;
  randomly pull $i(t)$ with regard to $P(i(t) = j)$;
  $B_{t+1} = B_t - c_{i(t)}$; $t = t + 1$;
end while

where $\gamma > 0$ is a constant value. Note that ε_t cannot be greater than 1. By using $M^*(B_t)$ and ε_t, KDE *randomly* selects the next arm to pull by choosing arm i with probability

$$P(i(t) = i) = (1 - \varepsilon_t) \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}} + \frac{\varepsilon_t}{K}. \tag{6.3}$$

The reason of using ε_t is that KDE can also randomly choose from the other arms, that are not involved in $M^*(B_t)$. By doing so, we can guarantee that the algorithm explores all the arms. However, the value of ε_t is decreased after each step, since as time passes by, we have more accurate estimation of arms, and thus, random exploration becomes less important. After the pull, it then updates the residual budget limit B_t (step 9).

The intuition behind KDE is the following: By repeatedly pulling an arm from the distribution formed by the current approximated best combination, the expected reward value that KDE receives at each time step follows the distribution of the approximated best combination of arms, that solves the corresponding unbounded knapsack problem. This indicates that the average performance of KDE efficiently converges to the optimal solution of the unbounded knapsack problem reduced from the budget–limited MAB model if the real value of the arms are known. It is easy to show that the optimal solution of this knapsack model forms the theoretical optimal policy of the budget–limited MAB. Given this, by using the density–ordered greedy method at each time step, KDE can achieve efficiently low regret bound, by converging to this theoretical optimal solution (see next section for more details).

6.1.2 Fractional KDE

Similar to KDE, fractional KDE also approximates the underlying unbounded knapsack problem at each time step t in order to determine the frequency of arms within the estimated best combination of arms. However, instead of using the density–ordered greedy algorithm, it uses a fractional relaxation based method to approximate the optimal solution of the unbounded knapsack. That is, similar to the case of the fractional KUBE, the following fractional unbounded knapsack is formed at each time step t:

$$\max \sum_{i=1}^{K} m_{i,t} \hat{\mu}_{i,n_{i,t}} \quad \text{s.t.} \quad \sum_{i=1}^{K} m_{i,t} c_i \leq B_t. \tag{6.4}$$

The optimal solution of this fractional problem is solely choosing arm $I^+(t)$, such that:

$$I^+(t) = \arg\max_j \frac{\hat{\mu}_{j,n_{j,t}}}{c_j}.$$

In other words, $I^{+}(t)$ denotes the arm with the highest expected reward density estimate. Given this, fractional KDE, depicted in Algorithm 6.2, can be described as follows: Similar to KDE, it first sets the value of ε_t (step 6). It then randomly chooses an arm to pull such that $I^{+}(t)$ is chosen with probability $(1-\varepsilon_t)$, and the others are chosen with probability $\frac{\varepsilon_t}{K}$ (steps $7-8$). Note that the fractional KDE can be regarded as the budget–limited version of the decreasing ε–greedy (see Section 2.2 for more details).

In what follows, we show that both KDE and its fractional counterpart achieve asymptotically optimal regret bounds. That is, we show that both algorithms achieve logarithmic regret bounds. According to Theorem 5.7, this implies that the regret bounds of the algorithm only differ from the best possible with a constant factor.

6.2 Performance Analysis

To provide an upper bound for the performance regret of KDE, we first state the following:

Lemma 6.1. *Let* $0 < d < d_{\min}$, $\gamma \geq \frac{56K}{3d^2}$, *and* $C = \frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}$, *where* K *is the number of arms. Suppose that the total budget size is* B. *Given this, if* T *denotes the total number of pulls of* KDE *then we have:*

$$\mathbf{E}[T] \geq \frac{B}{c_{I^*}} - \frac{C}{c_{I^*}} \sum_{\delta_j > 0} \delta_j \ln\left(\frac{B}{c_{\min}}\right) - \gamma \frac{\sum_j c_j}{K c^{I^*}} - \frac{C}{c_{I^*}} \sum_{\delta_j > 0} \delta_j - 1,$$

where $\mathbf{E}[T]$ *is the expected number of pulls using* KDE.

That is, the difference between $\frac{B}{c_{I^*}}$ and the number of pulls of KDE is at most $O\left(\ln\left(\frac{B}{c_{\min}}\right)\right)$. Let us first refresh some of the terms that are used: $i(t)$ is the arm pulled by KDE at time t, and when referring to a combination of arms $\{m_{j,t}\}$, $m_{j,t}$ is the number of pulls of arm j. In addition, recall that $I^{+}(t) = \arg\max_j \frac{\hat{\mu}_{j,n_{j,t}}}{c_j}$ denotes the arm with the highest estimated mean value density at time step t. In order to prove Lemma 6.1, we rely on the following lemmas:

Lemma 6.2. *Let* $0 < d < d_{\min}$, $\gamma \geq \frac{56K}{3d^2}$, *where* K *is the number of arms. For any* $t \geq \gamma$, *we get:*

$$P\left(I^{+}(t) = j, j \neq I^*\right) \leq \left(\frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}\right)\frac{1}{t}.$$

That is, the probability that the arm with the highest estimated mean value density is in fact suboptimal (i.e. it is not equal to I^*) at time step t is at most $O\left(\frac{1}{t}\right)$.

Lemma 6.3. *Suppose that at time t, the residual budget is B_t, and $M^*(B_t) = \{m_{i,t}\}$. Then:*

$$\frac{\sum_j m_{j,t} c_j}{\sum_j m_{j,t}} \;\leq\; c_{I^+(t)}\,.$$

That is, the weighted average cost of combination $\{m_{i,t}\}$ is at most $c_{I^+(t)}$.

Now, to prove Lemma 6.2, we will make use of the following version of the Bernstein's inequality for bounded random variables:

Theorem 6.4 (Bernstein's inequality - Theorem 10.2, Bubeck (2010))**.** *Let $X_1, X_2, \ldots, X_t$ denote the sequence of random variables with common range $[0,1]$, such that for any $1 \leq \tau \leq t$, we have $\mathbf{E}[X_\tau | X_1, \ldots, X_{\tau-1}] = \mu$, and $\sum_{\tau=1}^{t} \mathrm{Var}[X_\tau | X_1, \ldots, X_{\tau-1}] \leq v$ for some $v > 0$. Given this, for any $\delta \geq 0$, we have:*

$$P\left(\sum_{\tau=1}^{t} X_\tau \geq \mathbf{E}\left[\sum_{\tau=1}^{t} X_\tau\right] + \delta\right) \leq \exp\left\{-\frac{\delta^2}{2v + \frac{2\delta}{3}}\right\}, \tag{6.5}$$

$$P\left(\sum_{\tau=1}^{t} X_\tau \leq \mathbf{E}\left[\sum_{\tau=1}^{t} X_\tau\right] - \delta\right) \leq \exp\left\{-\frac{\delta^2}{2v + \frac{2\delta}{3}}\right\}. \tag{6.6}$$

The proof can be found, for example, in Bubeck (2010).

Proof of Lemma 6.2. Let:

$$x_t = \frac{1}{2K} \sum_{\tau=1}^{t} \varepsilon_\tau\,.$$

Now, we first show that:

$$P\left(I^+(t) = j, j \neq I^*\right) \;\leq\; \frac{\varepsilon_t}{K} + 2x_t \exp\left\{-\frac{3x_t}{14}\right\} + \frac{4}{d^2} \exp\left\{-\frac{d^2 \lfloor x_t \rfloor}{2}\right\}. \tag{6.7}$$

This inequality can be proved by a standard application of the Chernoff–Hoeffding (Theorem 4.2) and Bernstein's inequality, as for Theorem 3 in Auer *et al.* (2002). In particular, we have:

$$P\left(I^+(t) = j, j \neq I^*\right) \;\leq\; \frac{\varepsilon_t}{K} + P\left(\frac{\hat{\mu}_{j,n_{j,t}}}{c_j} \geq \frac{\hat{\mu}_{I^*,n_{I^*,t}}}{c_{I^*}}\right). \tag{6.8}$$

Similar to the proof of Equation 4.11 in Chapter 4, we bound the second term of the right hand side of Equation 6.8 as follows:

$$P\left(\frac{\hat{\mu}_{j,n_{j,t}}}{c_j} \geq \frac{\hat{\mu}_{I^*,n_{I^*,t}}}{c_{I^*}}\right) \leq P\left(\frac{\hat{\mu}_{j,n_{j,t}}}{c_j} \geq \frac{\mu_{j,n_{j,t}}}{c_j} + \frac{d_j}{2}\right) + P\left(\frac{\hat{\mu}_{I^*,n_{I^*,t}}}{c_{I^*}} \leq \frac{\mu_{I^*,n_{I^*,t}}}{c_{I^*}} - \frac{d_j}{2}\right). \tag{6.9}$$

The analysis for both terms on the right hand side is the same. Thus, from now on we focus on the first term. Let $n^{\mathrm{R}}_{j,t}$ denote the number of time steps in which arm j

was randomly chosen from the uniform distribution (and not from the current estimated best combination $\{m^*_{j,t}\}$) in the first t time steps. Given this, we have:

$$\begin{aligned} P\left(\frac{\hat{\mu}_{j,n_{j,t}}}{c_j} \geq \frac{\mu_{j,n_{j,t}}}{c_j} + \frac{d_j}{2}\right) &= \sum_{n=1}^{t} P\left(n_{j,t} = n, \frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}\right) \\ &= \sum_{n=1}^{t} P\left(n_{j,t} = n \middle| \frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}\right) P\left(\frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}\right) \\ &\leq \sum_{n=1}^{t} P\left(n_{j,t} = n \middle| \frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}\right) \mathrm{e}^{\frac{-nd_j^2}{2}}. \end{aligned} \tag{6.10}$$

The last inequality is obtained from the Chernoff–Hoeffding inequallity (Theorem 4.2). By using elementary algebra, it is easy to prove that for any $\kappa > 0$:

$$\sum_{n=x+1}^{\infty} \mathrm{e}^{-n\kappa} \leq \frac{1}{\kappa}\mathrm{e}^{-\kappa x}.$$

Substituting this into Equation 6.10 we obtain:

$$\begin{aligned} P\left(\frac{\hat{\mu}_{j,n_{j,t}}}{c_j} \geq \frac{\mu_{j,n_{j,t}}}{c_j} + \frac{d_j}{2}\right) \leq & \\ &\leq \sum_{n=1}^{\lfloor x_t \rfloor} P\left(n_{j,t} = n \middle| \frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}\right) + \frac{2}{d_j^2} \exp\left\{\frac{-d_j^2 \lfloor x_t \rfloor}{2}\right\} \\ &\leq \sum_{n=1}^{\lfloor x_t \rfloor} P\left(n^{\mathrm{R}}_{j,t} \leq n \middle| \frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}\right) + \frac{2}{d_j^2} \exp\left\{\frac{-d_j^2 \lfloor x_t \rfloor}{2}\right\} \\ &\leq \sum_{n=1}^{\lfloor x_t \rfloor} P\left(n^{\mathrm{R}}_{j,t} \leq n\right) + \frac{2}{d_j^2} \exp\left\{\frac{-d_j^2 \lfloor x_t \rfloor}{2}\right\} \\ &\leq x_t P\left(n^{\mathrm{R}}_{j,t} \leq x_t\right) + \frac{2}{d_j^2} \exp\left\{\frac{-d_j^2 \lfloor x_t \rfloor}{2}\right\}. \end{aligned} \tag{6.11}$$

Note that in the last two inequalities, we drop the conditioning to $\frac{\hat{\mu}_{j,n}}{c_j} \geq \frac{\mu_{j,n}}{c_j} + \frac{d_j}{2}$, since $n^{\mathrm{R}}_{j,t}$ only considers the cases when arm j is randomly chosen to be pulled from a uniform distribution, independently from the previous choices of the KDE algorithm. We now estimate $P\left(n^{\mathrm{R}}_{j,t} \leq x_t\right)$ as follows:

$$\mathbf{E}\left[n^{\mathrm{R}}_{j,t}\right] = \frac{1}{K}\sum_{\tau=1}^{t} \varepsilon_t = 2x_t,$$

and

$$\mathrm{Var}\left[n^{\mathrm{R}}_{j,t}\right] = \sum_{\tau=1}^{t} \frac{\varepsilon_t}{K}\left(1 - \frac{\varepsilon_t}{K}\right) \leq \frac{1}{K}\sum_{\tau=1}^{t} \varepsilon_t = 2x_t.$$

As a result, from Bernstein's inequality (Equation 6.6 from Theorem 6.4) we get:

$$P\left(n_{j,t}^{\mathrm{R}} \leq x_t\right) = P\left(n_{j,t}^{\mathrm{R}} \leq \mathbf{E}\left[n_{j,t}^{\mathrm{R}}\right] - x_t\right) \leq \exp\left\{\frac{-3x_t}{14}\right\}. \tag{6.12}$$

Substituting this into Equation 6.11, and combining with Equations 6.8 and 6.9, we obtain Equation 6.7 (note that a similar analysis has to be done for the second term on the right hand side of Equation 6.9).

Now, since the right hand side of Equation 6.7 is a monotone decreasing function, we lower bound the value of x_t, in order to upper bound $P\left(I^+\left(t\right) = j, j \neq I^*\right)$. Recall that $\varepsilon_\tau = \min\left\{1, \frac{\gamma}{\tau}\right\}$, so we can write:

$$x_t = \frac{1}{2K}\sum_{\tau=1}^{t}\varepsilon_\tau \;\geq\; \sum_{\tau=\lfloor\gamma\rfloor+1}^{t}\varepsilon_\tau \;\geq\; \frac{\gamma}{2K}\ln\left(\frac{t}{\gamma}\right).$$

Regarding the second term on the right–hand side of Equation 6.7, along with the above, note that $\gamma \geq \frac{56K}{3d^2}$ and $d \leq 1$, so $\left(\frac{\gamma}{t}\right)^{\frac{3\gamma}{28K}} \leq \left(\frac{\gamma}{t}\right)^2$. Therefore:

$$2x_t \exp\left\{-\frac{3x_t}{14}\right\} \;\leq\; \frac{\gamma}{K}\ln\left(\frac{t}{\gamma}\right)\left(\frac{\gamma}{t}\right)^{\frac{3\gamma}{28K}} \;\leq\; \frac{\gamma}{K}\ln\left(\frac{t}{\gamma}\right)\left(\frac{\gamma}{t}\right)^2 \;\leq\; \frac{\gamma^2}{K}\frac{1}{t}. \tag{6.13}$$

Similarly, for the third term on the on the right–hand side of Equation 6.7, we have:

$$\frac{4}{d^2}\exp\left\{-\frac{d^2\left\lfloor x_t\right\rfloor}{2}\right\} \;\leq\; \frac{4}{d^2}\exp\left(\frac{d^2}{2}\right)\left(\frac{\gamma}{t}\right)^{\frac{d^2\gamma}{4K}} \;\leq\; \frac{4\gamma}{d^2}e^{\frac{1}{2}}\frac{1}{t}. \tag{6.14}$$

In addition, since $t \geq \gamma$, we have $\varepsilon_t = \frac{\gamma}{t}$. By using this and Equations 6.13 and 6.14, we get:

$$P\left(I^+\left(t\right) = j, j \neq I^*\right) \;\leq\; \frac{\gamma^2}{K}\frac{1}{t} + \left(\frac{\gamma}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}\right)\frac{1}{t}, \tag{6.15}$$

which concludes the proof. □

Proof of Lemma 6.3. Here, we consider two cases. In the first one, suppose that the combination of arms that the density–ordered greedy algorithm returns contains only one arm, namely $I^+\left(t\right)$ (i.e. $\{m_{i,t}\}$ is >0 for $m_{I^+(t),t}$ only). In this case, $\sum_j m_{j,t}c_j = m_{I^+(t),t}c_{I^+(t)}$ and $\sum_j m_{j,t} = m_{I^+(t),t}$. Thus:

$$\frac{\sum_j m_{j,t}c_j}{\sum_j m_{j,t}} = \frac{m_{I^+(t),t}c_{I^+(t)}}{m_{I^+(t),t}} = c_{I^+(t)}.$$

Now consider the second case where $\{m_{i,t}\}$ is > 0 for more than one arm. Recall that the density–ordered greedy algorithm repeatedly adds arm $I^+\left(t\right)$ into the combination until it is no longer feasible, and then it adds the feasible arm with the next highest

estimated density. This implies that:

$$\sum_j m_{j,t} \geq m_{I^+(t)} + 1 = \left\lfloor \frac{B_t}{c_{I^+(t)}} \right\rfloor + 1 \geq \frac{B_t}{c_{I^+(t)}} \,. \tag{6.16}$$

By definition $\sum_j m_{j,t} c_j \leq B_t$. From this and Equation 6.16, we get:

$$\frac{\sum_j m_{j,t} c_j}{\sum_j m_{j,t}} \leq \frac{B_t}{\frac{B_t}{c_{I^+(t)}}} = c_{I^+(t)} \,.$$

Thus we have proved the two cases of the combination of arms containing a single arm or two or more arms. □

Based on Lemmas 6.2 and 6.3, Lemma 6.1 can be proved as follows:

Proof of Lemma 6.1. Since KDE pulls arms until none are feasible, by definition:

$$P\left(\sum_{t=1}^{T} c_{i(t)} > B - c_{\min}\right) = 1.$$

Given this, taking the expectation of $\sum_{t=1}^{T} c_{i(t)}$ over T and $\{m_{j,t}\}$ (i.e. the set of $i\,(t)$) gives:

$$\mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T} c_{i(t)}\right] = \mathbf{E}_T\left[\sum_{t=1}^{T} \mathbf{E}_{i(t)}\left[c_{i(t)}\right]\right] > B - c_{\min} \,. \tag{6.17}$$

Now, for all $0 < t \leq T$, we have:

$$\mathbf{E}_{i(t)}\left[c_{i(t)}\right] = \sum_{\{m_{j,t}\}} \frac{\sum_j m_j c_j}{\sum_j m_j} P\left(M^*\left(B_t\right) = \{m_{j,t}\}\right) \leq \sum_{\{m_{j,t}\}} c_{I^+(t)} P\left(M^*\left(B_t\right) = \{m_{j,t}\}\right). \tag{6.18}$$

where the second inequality comes from Lemma 6.3.

Now consider two cases of $t \leq \lfloor \gamma \rfloor$ and $t > \lfloor \gamma \rfloor$. First, for $t \leq \lfloor \gamma \rfloor$, we have:

$$\mathbf{E}_{i(t)}\left[c_{i(t)}\right] = \frac{\sum_j c_j}{K} \,. \tag{6.19}$$

because in the first $\lfloor \gamma \rfloor$ steps, KDE randomly pulls each arm j with probability $\frac{1}{K}$ (see Algorithm 6.1 for more details).

Second, suppose that $t > \lfloor \gamma \rfloor$. Recall that the density–ordered greedy algorithm first adds the arm with the highest density estimate (i.e. $I^+\left(t\right)$) into the combination of arms. Given this, we can group the possible combinations together so that combinations that have the same value of $I^+\left(t\right)$ are in the same group. By doing this, Equation 6.18 can

be restated as follows:

$$\mathbf{E}_{i(t)}\left[c_{i(t)}\right] \leq \sum_{\{m_{j,t}\}} c_{I(t)} P\left(M^*\left(B_t\right) = \{m_{j,t}\}\right) = \sum_j c_j P\left(I^+\left(t\right) = j\right) \leq$$
$$\leq c_{I^*} + \sum_{\delta_j > 0} \delta_j \left(\frac{\gamma + \gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}\right) \frac{1}{t}\,. \tag{6.20}$$

Note that the last inequality comes from Lemma 6.2. In addition, since δ_j can be smaller than 0 for some arm j, we can further upper bound by only considering arms with $\delta_j > 0$.

Substituting Equations 6.19 and 6.20 into Equation 6.17, and using $C = \frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}$, gives:

$$\begin{aligned} B - c_{\min} &< \sum_{t=1}^{\lfloor\gamma\rfloor} \frac{\sum_j c_j}{K} + \mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\left(c_{I^*} + \sum_{\delta_j>0}\delta_j C\frac{1}{t}\right)\right] \\ &\leq \gamma\frac{\sum_j c_j}{K} + c_{I^*}\mathbf{E}\left[T\right] + \sum_{\delta_j>0}\delta_j C\mathbf{E}_T\left[\sum_{t=1}^{T}\frac{1}{t}\right] \\ &\leq \gamma\frac{\sum_j c_j}{K} + c_{I^*}\mathbf{E}\left[T\right] + C\sum_{\delta_j>0}\delta_j\mathbf{E}_T\left[\ln\left(T\right)+1\right] \\ &\leq \gamma\frac{\sum_j c_j}{K} + c_{I^*}\mathbf{E}\left[T\right] + C\sum_{\delta_j>0}\delta_j\left(\ln\left(\frac{B}{c_{\min}}\right)+1\right). \end{aligned} \tag{6.21}$$

The last two inequalities come from the fact that $\sum_{t=1}^{T}\frac{1}{t} \leq \ln\left(T\right)+1$ and that $T \leq \frac{B}{c_{\min}}$ for any possible T. Keeping $c_{I^*}\mathbf{E}\left[T\right]$ only on the right hand side of Equation 6.21 and dividing the both sides with c_{I^*} gives the stated inequality (n.b. $\frac{c_{\min}}{c_{I^*}} \leq 1$). □

Note that if we relax the budget–limited MAB problem so that the number of pulls can be fractional, then it is easy to show that the optimal pulling policy of this relaxed model is to repeatedly pull arm I^* only. In this case, $\frac{B}{c_{I^*}}$ is the number of pulls of this optimal policy. Given this, Theorem 6.1 indicates that the number of pulls that KDE produces does not significantly differ from that of the optimal policy of the fractional budget–limited MAB (i.e. the difference is a logarithmic function of the number of pulls). From Lemma 6.1, we get:

Theorem 6.5. *Let* $0 < d < d_{\min}$, $\gamma \geq \frac{56K}{3d^2}$, *and* $C = \frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}$, *where* K *is the number of arms. Given this, for any budget size* $B > 0$, *the performance regret of* KDE *is at most*

$$\left(C\sum_{\Delta_j>0}\Delta_j + C\frac{\sum_{\delta_j>0}\delta_j}{c_{I^*}} + \frac{\gamma}{K}\sum_{\Delta_j>0}\Delta_j + \left(\frac{c_{\max}}{c_{\min}}\right)^2 + 1\right)\ln\left(\frac{B}{c_{\min}}\right)$$
$$+\gamma\left(\frac{\sum_j c_j}{Kc_{I^*}} + \frac{\sum_{\Delta_i>0}\Delta_i}{K}\right) + C\sum_{\Delta_j>0}\Delta_j + \left(\frac{c_{\max}}{c_{\min}}\right)^2 + 2\,.$$

To prove this theorem, we first prove the following lemmas:

Lemma 6.6. *Suppose that at time step t, the residual budget is B_t, and $M^*(B_t) = \{m_{i,t}\}$. That is, the density–ordered greedy approach that* KDE *uses returns $\{m_{i,t}\}$ as the best combination of arms. Given this, we get:*

$$\mu_{I^*} - \frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}} \leq \Delta_{I^+(t)} + \left(\left(\frac{c_{\max}}{c_{\min}}\right)^2 + 1\right)\frac{c_{\min}}{B_t}.$$

Lemma 6.7. *Suppose that* KDE *pulls the arms T times, and that at each time step t, the residual budget is B_t (note that here $B_1 = B$). Given this, for any $0 < t \leq T$, we have:*

$$\frac{c_{\min}}{B_t} \leq \frac{1}{T-t+1}.$$

Proof of Lemma 6.6. Without loss of generality, we assume that the density–ordered greedy approach adds the arms into combination $\{m_{i,t}\}$ in the order of $\{1, 2, \ldots, K\}$. That is, here $I^+(t) = 1$. Due to the nature of the density–ordered greedy method, it is easy to show that $m_{1,t} = \left\lfloor \frac{B_t}{c_1} \right\rfloor$. It is also easy to show that $\sum_{j=1}^{K} m_{j,t} \leq \frac{B_t}{c_{\min}}$, since we can achieve the maximal number of pulls by repeatedly pulling the arm with minimal cost. Given this, we get:

$$\frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}} \geq \frac{\left\lfloor \frac{B_t}{c_1} \right\rfloor \mu_1}{\sum_j m_{j,t}} \geq \frac{\frac{B_t}{c_1}\mu_1 - 1}{\sum_j m_{j,t}} \geq \frac{\frac{B_t}{c_1}\mu_1}{\sum_j m_{j,t}} - \frac{c_{\min}}{B_t}. \tag{6.22}$$

Recall that after repeatedly adding arm 1 to $\{m_{i,t}\}$, the residual budget is $B_t - m_{1,t}c_1 < c_1$, otherwise we can still add arm 1 at least once to $\{m_{i,t}\}$. It is easy to see that the maximum number of arms we can add to $\{m_{i,t}\}$ with respect to this residual budget is when we add only the arm with the smallest pulling cost. Given this, we have:

$$\sum_{j=2}^{K} m_{j,t} \leq \frac{c_1}{c_{\min}} \leq \frac{c_{\max}}{c_{\min}}.$$

That is, we get:

$$\frac{\frac{B_t}{c_1}\mu_1}{\sum_j m_{j,t}} \geq \frac{\frac{B_t}{c_1}\mu_1}{\frac{B_t}{c_1} + \frac{c_{\max}}{c_{\min}}} = \sigma. \tag{6.23}$$

By using Equations 6.22 and 6.23, and that $\mu_1 \leq 1$, we obtain the following:

$$\begin{aligned}
\mu_{I^*} - \frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}} &\leq \mu_{I^*} - \sigma + \frac{c_{\min}}{B_t} = \mu_{I^*} - \mu_1 + \frac{\frac{c_{\max}}{c_{\min}}}{\frac{B_t}{c_1} + \frac{c_{\max}}{c_{\min}}}\mu_1 + \frac{c_{\min}}{B_t} \\
&\leq \Delta_1 + \frac{\frac{c_{\max}}{c_{\min}}}{\frac{B_t}{c_1}} + \frac{c_{\min}}{B_t} \leq \Delta_{I^+(t)} + \frac{\frac{c_{\max}^2}{c_{\min}}}{B_t} + \frac{c_{\min}}{B_t}
\end{aligned}$$

where the last inequality is obtained from $c_1 \leq c_{\max}$ and $I^+(t) = 1$. This concludes the proof. □

Proof of Lemma 6.7. At the beginning of time step t, the residual budget is B_t. Since the total number of pulls is T, with respect to B_t, KDE can still achieves $T-t+1$ pulls (including the pull at time step t). Given this, we have:

$$B_t \geq c_{i(t)} + c_{i(t+1)} + \cdots + c_{i(T)} \geq (T-t+1)\, c_{\min}.$$

which directly implies the inequality in Lemma 6.7. □

Now we prove Theorem 6.5 as follows:

Proof of Theorem 6.5. Recall that $\mathbf{E}\left[G^B(A^*)\right]$ denotes the expected performance of the theoretical optimal policy. It is obvious that $\mathbf{E}\left[G^B(A^*)\right] \leq \frac{B\mu_{I^*}}{c_{I^*}}$, since the latter is the optimal solution of the fractional budget–limited MAB problem. Given this, we have the following:

$$R^B(\text{KDE}) = \mathbf{E}\left[G^B(A^*)\right] - \mathbf{E}\left[G^B(\text{KDE})\right] \leq \frac{B\mu_{I^*}}{c_{I^*}} - \mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T}\mu_{i(t)}\right] \Rightarrow$$

$$\Rightarrow R^B(\text{KDE}) \leq \frac{B\mu_{I^*}}{c_{I^*}} - \mathbf{E}_T\left[\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right] = \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right]. \tag{6.24}$$

Now, consider $\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]$. According to the definition of KDE in Section 6.1.1, we get:

$$\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right] = (1-\varepsilon_t)\sum_{\{m_{j,t}\}}\frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}}P\left(M^*(B_t) = \{m_{j,t}\}\right) + \frac{\varepsilon_t\sum_i \mu_i}{K}.$$

Substituting this into Equation 6.24, we have:

$$\begin{aligned}
R^B(\text{KDE}) &\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T}\left((1-\varepsilon_t)\sum_{\{m_{j,t}\}}\frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}}P\left(M^*(B_t) = \{m_{j,t}\}\right) + \frac{\varepsilon_t\sum_i\mu_i}{K}\right)\right] \\
&\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - T\mu_{I^*} + \sum_{t=1}^{T}\left\{(1-\varepsilon_t)\left(\mu_{I^*} - \sum_{\{m_{j,t}\}}\frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}}P\left(M^*(B_t) = \{m_{j,t}\}\right)\right) + \right.\right. \\
&\quad \left.\left. + \varepsilon_t\left(\mu_{I^*} - \frac{\sum_i\mu_i}{K}\right)\right\}\right] \\
&\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - T\mu_{I^*}\right] + \mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\left(\mu_{I^*} - \sum_{\{m_{j,t}\}}\frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}}P\left(M^*(B_t) = \{m_{j,t}\}\right)\right)\right] +
\end{aligned}$$

$$
\begin{aligned}
&+ \frac{\sum_i (\mu_{I^*} - \mu_i)}{K} \mathbf{E}_T \left[\sum_{t=1}^{T} \varepsilon_t \right] \\
&\leq \mathbf{E}_T \left[\frac{B}{c_{I^*}} - T \right] \mu_{I^*} + \mathbf{E}_T \left[\sum_{t=\lfloor \gamma \rfloor + 1}^{T} \sum_{\{m_{j,t}\}} \left(\mu_{I^*} - \frac{\sum_j m_{j,t} \mu_j}{\sum_j m_{j,t}} \right) P\left(M^*\left(B_t\right) = \{m_{j,t}\} \right) \right] + \\
&+ \frac{\sum_{\Delta_i > 0} \Delta_i}{K} \mathbf{E}_T \left[\sum_{t=1}^{T} \varepsilon_t \right] . \qquad (6.25)
\end{aligned}
$$

The third inequality is obtained from the fact that $\varepsilon_t = 1$ if $t \leq \lfloor \gamma \rfloor$ and $\varepsilon_t < 1$ otherwise. The last inequality also holds since $\sum_i (\mu_{I^*} - \mu_i) \leq \sum_{\Delta_i > 0} \Delta_i$. Note that here Δ_j can be smaller than 0 for some arm j, thus, we can further upper bound by only considering arms with $\Delta_j > 0$.

In what follows, we provide upper bounds for each of the three terms on the right hand side of Equation 6.25. In so doing, we first use Lemma 6.1 to obtain the following:

$$
\mathbf{E}_T \left[\frac{B}{c_{I^*}} - T \right] \mu_{I^*} \leq \frac{C}{c_{I^*}} \sum_{\delta_j > 0} \delta_j \ln \left(\frac{B}{c_{\min}} \right) + \gamma \frac{\sum_j c_j}{K c_{I^*}} + \frac{C}{c_{I^*}} \sum_{\delta_j > 0} \delta_j + 1 \,. \qquad (6.26)
$$

Here we exploit the fact that $\mu_{I^*} \leq 1$. Now we turn to bound the second term on the right hand side of Equation 6.25. From Lemma 6.6 we have:

$$
\mu_{I^*} - \frac{\sum_j m_{j,t} \mu_j}{\sum_j m_{j,t}} \leq \Delta_{I^+(t)} + \left(\left(\frac{c_{\max}}{c_{\min}} \right)^2 + 1 \right) \frac{c_{\min}}{B_t} \,.
$$

This implies the following:

$$
\begin{aligned}
&\mathbf{E}_T \left[\sum_{t=\lfloor \gamma \rfloor + 1}^{T} \sum_{\{m_{j,t}\}} \left(\mu_{I^*} - \frac{\sum_j m_{j,t} \mu_j}{\sum_j m_{j,t}} \right) P\left(M^*\left(B_t\right) = \{m_{j,t}\} \right) \right] \leq \\
&\quad \leq \mathbf{E}_T \left[\sum_{t=\lfloor \gamma \rfloor + 1}^{T} \left\{ \sum_{\{m_{j,t}\}} \Delta_{I^+(t)} P\left(M^*\left(B_t\right) = \{m_{j,t}\} \right) + \left(\left(\frac{c_{\max}}{c_{\min}} \right)^2 + 1 \right) \frac{c_{\min}}{B_t} \right\} \right] . \qquad (6.27)
\end{aligned}
$$

Now, by grouping the possible sets of $\{m_{j,t}\}$ together so that the combinations with the same $I^+(t)$ belongs to the same group, we can reformalise Equation 6.27 as follows:

$$
\begin{aligned}
&\mathbf{E}_T \left[\sum_{t=\lfloor \gamma \rfloor + 1}^{T} \sum_{\{m_{j,t}\}} \left(\mu_{I^*} - \frac{\sum_j m_{j,t} \mu_j}{\sum_j m_{j,t}} \right) P\left(M^*\left(B_t\right) = \{m_{j,t}\} \right) \right] \leq \\
&\quad \leq \mathbf{E}_T \left[\sum_{t=\lfloor \gamma \rfloor + 1}^{T} \sum_{j=1}^{K} \Delta_j P\left(I^+(t) = j \right) \right] + \mathbf{E}_T \left[\sum_{t=\lfloor \gamma \rfloor + 1}^{T} \left(\left(\frac{c_{\max}}{c_{\min}} \right)^2 + 1 \right) \frac{c_{\min}}{B_t} \right] . \qquad (6.28)
\end{aligned}
$$

Let $C = \frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}$. From Lemma 6.2, we have:

$$\begin{aligned}\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\sum_{j=1}^{K}\Delta_j P\left(I^+(t)=j\right)\right] &\leq \mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\sum_{j=1}^{K}\Delta_j C\frac{1}{t}\right] \\ &\leq C\sum_{j=1}^{K}\Delta_j\mathbf{E}_T\left[\ln(T)+1\right] \leq C\sum_{j=1}^{K}\Delta_j\left(\ln\left(\frac{B}{c_{\min}}\right)+1\right).\end{aligned} \tag{6.29}$$

The last inequality holds since $T \leq \frac{B}{c_{\min}}$ with probability 1. Next, by Lemma 6.7:

$$\begin{aligned}\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\left(\left(\frac{c_{\max}}{c_{\min}}\right)^2+1\right)\frac{c_{\min}}{B_t}\right] &\leq \left(\left(\frac{c_{\max}}{c_{\min}}\right)^2+1\right)\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\frac{1}{T-t+1}\right] \\ &\leq \left(\left(\frac{c_{\max}}{c_{\min}}\right)^2+1\right)\left(\mathbf{E}_T\left[\ln(T)\right]+1\right) \\ &\leq \left(\left(\frac{c_{\max}}{c_{\min}}\right)^2+1\right)\left(\ln\left(\frac{B}{c_{\min}}\right)+1\right).\end{aligned} \tag{6.30}$$

The second inequality is obtained from the fact that $\sum_{t=\lfloor\gamma\rfloor+1}^{T}\frac{1}{t} \leq \ln(T)+1$. Subtituting Equations 6.29 and 6.30 into Equation 6.28 implies that:

$$\begin{aligned}&\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\sum_{\{m_{j,t}\}}\left(\mu_{I^*}-\frac{\sum_j m_{j,t}\mu_j}{\sum_j m_{j,t}}\right)P\left(M^*(B_t)=\{m_{j,t}\}\right)\right] \leq \\ &\leq \left(C\sum_{j=1}^{K}\Delta_j+\left(\frac{c_{\max}}{c_{\min}}\right)^2+1\right)\left(\ln\left(\frac{B}{c_{\min}}\right)+1\right).\end{aligned} \tag{6.31}$$

As the last step of the proof, we now bound the third term on the right hand side of Equation 6.25. Recall that $\varepsilon_t = 1$ if $t \leq \lfloor\gamma\rfloor$ and $\varepsilon_t = \frac{\gamma}{t}$ otherwise. Given this, we have:

$$\begin{aligned}\frac{\sum_{\Delta_i>0}\Delta_i}{K}\mathbf{E}_T\left[\sum_{t=1}^{T}\varepsilon_t\right] &\leq \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma+\frac{\sum_{\Delta_i>0}\Delta_i}{K}\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\frac{\gamma}{t}\right] \\ &\leq \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma+\frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma\mathbf{E}_T\left[\ln(T)\right] \\ &\leq \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma+\frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma\ln\left(\frac{B}{c_{\min}}\right).\end{aligned} \tag{6.32}$$

Now, by subtituting Equations 6.26, 6.31, and 6.32 into Equation 6.25, we get the stated bound for the performance regret of KDE . □

Similarly, the regret of fractional KDE is bounded as follows:

Theorem 6.8. *Let $0 < d < d_{\min}$, $\gamma \geq \frac{56K}{3d^2}$, and $C = \frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}$, where K is the number of arms. Given this, for any budget size $B > 0$, the performance regret of the*

fractional KDE *is at most*

$$\left(C\sum_{\Delta_j>0}\Delta_j + C\frac{\sum_{\delta_j>0}\delta_j}{c_{I^*}} + \frac{\gamma}{K}\sum_{\Delta_j>0}\Delta_j\right)\ln\left(\frac{B}{c_{\min}}\right) + \gamma\left(\frac{\sum_j c_j}{Kc_{I^*}} + \frac{\sum_{\Delta_i>0}\Delta_i}{K}\right) + C\sum_{\Delta_j>0}\Delta_j + 1\,.$$

We now turn to prove Theorem 6.33, which provides a theoretical upper bound for the regret of the fractional KDE.

Proof of Theorem 6.33. We follow the concept that is similar to the proof of Theorem 6.5. In particular, analogous to Lemma 6.2, we can easily show that if $0 < d < d_{\min}$, and $\gamma \geq \frac{56K}{3d^2}$, where K is the number of arms, for any $t \geq \gamma$, we get:

$$P\left(I^+(t) = j, j \neq I^*\right) \leq \left(\frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}\right)\frac{1}{t}\,. \tag{6.33}$$

In the next step, we calculate $\mathbf{E}[T]$, where T is the number of pulls within the fractional KDE. In so doing, consider two cases of $t \leq \lfloor\gamma\rfloor$ and $t > \lfloor\gamma\rfloor$. First, for $t \leq \lfloor\gamma\rfloor$, we have:

$$\mathbf{E}_{i(t)}\left[c_{i(t)}\right] = \frac{\sum_j c_j}{K}\,. \tag{6.34}$$

because in the first $\lfloor\gamma\rfloor$ steps, fractional KDE randomly pulls each arm j with probability $\frac{1}{K}$ (see Algorithm 6.2 for more details).

Second, suppose that $t > \lfloor\gamma\rfloor$. From Equation 6.33, we have:

$$\mathbf{E}_{i(t)}\left[c_{i(t)}\right] = \sum_j c_j P\left(I^+(t) = j\right) \leq c_{I^*} + \sum_{\delta_j>0}\delta_j\left(\frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}\right)\frac{1}{t}\,. \tag{6.35}$$

Since fractional KDE runs until it is not feasible to pull any arms, Equation 6.17 also holds:

$$\mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T} c_{i(t)}\right] = \mathbf{E}_T\left[\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[c_{i(t)}\right]\right] > B - c_{\min}\,. \tag{6.36}$$

Similar to the case of KDE, from Equations 6.34, 6.35, and 6.36, we get:

$$\mathbf{E}[T] \geq \frac{B}{c_{I^*}} - \frac{C}{c_{I^*}}\sum_{\delta_j>0}\delta_j\ln\left(\frac{B}{c_{\min}}\right) - \gamma\frac{\sum_j c_j}{Kc^{I^*}} - \frac{C}{c_{I^*}}\sum_{\delta_j>0}\delta_j - 1. \tag{6.37}$$

Note that this equation is analogous to Lemma 6.1. We now estimate the performance regret of the fractional KDE as follows. It is easy to show that

$$R^B\left(\text{fractional KDE}\right) \leq \frac{B\mu_{I^*}}{c_{I^*}} - \mathbf{E}_{T,\{i(t)\}}\left[\sum_{t=1}^{T}\mu_{i(t)}\right] \Rightarrow$$
$$\Rightarrow R^B\left(\text{fractional KDE}\right) \leq \frac{B\mu_{I^*}}{c_{I^*}} - \mathbf{E}_T\left[\sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right] = \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T}\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]\right]. \tag{6.38}$$

Consider $\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right]$. According to the definition of fractional KDE in Section 6.1.2, we get:

$$\mathbf{E}_{i(t)}\left[\mu_{i(t)}\right] = (1-\varepsilon_t)\sum_{j=1}^{K}\mu_j P\left(I^+(t)=j\right) + \frac{\varepsilon_t\sum_i\mu_i}{K}. \tag{6.39}$$

Substituting this into Equation 6.38, we have:

$$\begin{aligned}
R^B\left(\text{fractional KDE}\right) &\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - \sum_{t=1}^{T}\left((1-\varepsilon_t)\sum_{j=1}^{K}\mu_j P\left(I^+(t)=j\right) + \frac{\varepsilon_t\sum_i\mu_i}{K}\right)\right] \\
&\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - T\mu_{I^*} + \sum_{t=1}^{T}\left\{(1-\varepsilon_t)\left(\mu_{I^*} - \sum_{j=1}^{K}\mu_j P\left(I^+(t)=j\right)\right) + \right.\right. \\
&\quad \left.\left. + \varepsilon_t\left(\mu_{I^*} - \frac{\sum_i\mu_i}{K}\right)\right\}\right] \\
&\leq \mathbf{E}_T\left[\frac{B\mu_{I^*}}{c_{I^*}} - T\mu_{I^*}\right] + \mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\left(\mu_{I^*} - \sum_{j=1}^{K}\mu_j P\left(I^+(t)=j\right)\right)\right] + \\
&\quad + \frac{\sum_i\left(\mu_{I^*}-\mu_i\right)}{K}\mathbf{E}_T\left[\sum_{t=1}^{T}\varepsilon_t\right] \\
&\leq \mathbf{E}_T\left[\frac{B}{c_{I^*}} - T\right]\mu_{I^*} + \mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\sum_{j=1}^{K}\Delta_j P\left(I^+(t)=j\right)\right] + \\
&\quad + \frac{\sum_{\Delta_i>0}\Delta_i}{K}\mathbf{E}_T\left[\sum_{t=1}^{T}\varepsilon_t\right].
\end{aligned} \tag{6.40}$$

The third inequality is obtained from the fact that $\varepsilon_t = 1$ if $t \leq \lfloor\gamma\rfloor$ and $\varepsilon_t < 1$ otherwise. In addition, the last inequality holds since $\sum_i\left(\mu_{I^*}-\mu_i\right) \leq \sum_{\Delta_i>0}\Delta_i$. In what follows, we provide upper bounds for each of the three terms on the right hand side of Equation 6.40. In particular, from Equation 6.37, we have:

$$\mathbf{E}_T\left[\frac{B}{c_{I^*}} - T\right]\mu_{I^*} \leq \frac{C}{c_{I^*}}\sum_{\delta_j>0}\delta_j\ln\left(\frac{B}{c_{\min}}\right) + \gamma\frac{\sum_j c_j}{Kc_{I^*}} + \frac{C}{c_{I^*}}\sum_{\delta_j>0}\delta_j + 1. \tag{6.41}$$

Here we exploit the fact that $\mu_{I^*} \leq 1$. In addition, from Equation 6.33 we have:

$$\begin{aligned}
\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\sum_{j=1}^{K}\Delta_j P\left(I^+(t)=j\right)\right] &\leq \mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\sum_{j=1}^{K}\Delta_j C\frac{1}{t}\right] \\
&\leq C\sum_{j=1}^{K}\Delta_j \mathbf{E}_T\left[\ln(T)+1\right] \\
&\leq C\sum_{j=1}^{K}\Delta_j\left(\ln\left(\frac{B}{c_{\min}}\right)+1\right), \qquad (6.42)
\end{aligned}$$

where $C = \frac{\gamma+\gamma^2}{K} + \frac{4\gamma e^{\frac{1}{2}}}{d^2}$. Finally, recall that $\varepsilon_t = 1$ if $t \leq \lfloor\gamma\rfloor$ and $\varepsilon_t = \frac{\gamma}{t}$ otherwise. Given this, we have:

$$\begin{aligned}
\frac{\sum_{\Delta_i>0}\Delta_i}{K}\mathbf{E}_T\left[\sum_{t=1}^{T}\varepsilon_t\right] &\leq \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma + \frac{\sum_{\Delta_i>0}\Delta_i}{K}\mathbf{E}_T\left[\sum_{t=\lfloor\gamma\rfloor+1}^{T}\frac{\gamma}{t}\right] \\
&\leq \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma + \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma\mathbf{E}_T\left[\ln(T)\right] \\
&\leq \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma + \frac{\sum_{\Delta_i>0}\Delta_i}{K}\gamma\ln\left(\frac{B}{c_{\min}}\right). \qquad (6.43)
\end{aligned}$$

Combining Equations 6.41, 6.42, and 6.43, we get the requested upper bound. □

Theorems 6.5 and 6.8 imply that the performance regret of both KDE and its fractional counterpart is bounded by $O\left(\ln\left(\frac{B}{c_{\min}}\right)\right)$. Theorem 5.7 indicates that both algorithms follow the concept of asymptotic optimal convergence, and thus, they both satisfy Requirement 3.

From the computational aspect, since the underlying unbounded knapsack problem is the same as in the case of KUBE and fractional KUBE, it can easily be shown that the computational cost of KDE and its fractional counterpart is $O\left(\frac{BK\ln K}{c_{\min}}\right)$ and $O(BK)$, respectively. This implies that both algorithms satisfy Requirement 2 (i.e. low computational complexity). In addition, similar to the case of the UCB based approaches, fractional KDE outperforms KDE in terms of computational efficiency.

6.3 Summary

In this chapter, we focused on developing pulling algorithms that fulfil both the theoretical and empirical research requirements, providing a trade–off between the budget–limited ε–first and the KUBE approaches. To this end, we proposed two algorithms, KDE and fractional KDE, that combine the decreasing ε–greedy pulling policies with unbounded knapsack approximation methods. Similar to the KUBE approaches presented in the previous chapter, both KDE and fractional KDE use the current estimates

of the expected reward values to form an underlying unbounded knapsack problem at each time step t. To solve this knapsack problem, KDE uses a density–ordered greedy approximation approach, while fractional KUBE relies on a fractional relaxation technique. We showed that these algorithms provide efficient theoretical regret bounds that follow the concept of asymptotic optimal convergence; that is, they both efficiently satisfy Requirement 3. In more detail, we first provided a lower bound for the value of T, the number of pulls within KDE (Lemma 6.1). We then provided an $O(\ln B)$ upper bound for the regret of KDE (Theorem 6.5). In a similar vein, we also showed that fractional KDE achieves a logarithmic upper bound (Theorem 6.8). From Theorem 5.7, we can easily show that both algorithms achieve asymptotic optimal bounds, and thus, they satisfy Requirement 3.

In addition, since the KDE approaches follow the concept that is similar to the KUBE approaches in tackling the underlying unbounded knapsack, the computational complexity of the KDE approaches are similar to that of the KUBE algorithms. In particular, KDE has a $O(B(K + \ln K))$ computational cost, and its fractional counterpart achieves a reduced cost of $O(BK)$. That is, both algorithms have low computational cost, compared to the budget size B, and the number of arms K. This indicates that the algorithms fulfil Requirement 2 (i.e. computational feasibility).

So far, we have only analysed the proposed algorithms from the theoretical perspective, focusing on their fulfilment of Requirements 2 and 3. In the next chapter, we investigate the empirical efficiency of the algorithms by carrying out an experimental study within the domain of wireless sensor networks. With this study, we will demonstrate that the budget–limited ε–first approach shows efficient performance, and the KUBE algorithms perform poorly. Meanwhile, we will show that the KDE algorithms provide good performance, compared to that of the budget–limited ε–first approach. Thus, the KDE algorithms act as a good trade–off between the budget–limited ε–first and KUBE approaches by achieving good performance from both the theoretical and the experimental perspectives.

Chapter 7

Long–Term Information Collection in Wireless Sensor Networks

In each of the previous three chapters we have considered pulling algorithms, namely: budget–limited ε–first, KUBE, fractional KUBE, KDE, and fractional KDE, that are designed for budget–limited multi–armed bandits. In more detail, we have focused on the development of these algorithms against Requirements 2 and 3. That is, whether they are computationally feasible (Requirement 2), and achieve efficient finite–time regret bounds (Requirement 3). Within this chapter, we study the experimental performance quality (Requirement 1) of the algorithms, in order to investigate whether they fulfil this aspect of the research. In so doing, we apply the algorithms to the problem of long–term information collection in wireless sensor networks, which is one of the key research challenges within the WSN domain (see Section 1.2 for more detail). In addition, we show that by using the budget–limited MAB algorithms, we extend the state–of–the–art in terms of efficient long–term information collection within WSNs. In particular, we demonstrate that our budget–limited MAB approach outperforms USAC, a state–of–the–art method within the domain of information collection in WSNs (Padhy *et al.*, 2010).

To this end, we first revise the related work within the research domain of information collection in WSNs (Section 7.1), and we then formalise the problem of long–term information collection in Section 7.2. In particular, we decompose this problem into two sub–problems, namely (i) energy management; and (ii) maximal information throughput routing (see Section 1.3 for more detail). We then provide a budget–limited MAB approach for the former in Section 7.3, and an optimal decentralised routing algorithm

for the latter in Section 7.4. The empirical performance of the aforementioned budget–limited MAB algorithms are then evaluated in Section 7.5. In addition, we also compare their performance with that of USAC.

7.1 Related Work

In this section, we provide an overview of existing research studies against which this application of our work is positioned. In order to do so, in the first part of the section (Sections 7.1.1– 7.1.4) we discuss previous work on data collection of WSNs from four different aspects, namely: data sampling, information content valuation, information–centric routing, and energy management. Within these areas, we highlight the limitations of each of the proposed methods, motivating the solution we present in this chapter.

In more detail, in Section 7.1.1, we first describe some of the most commonly used adaptive sampling methods that have been developed for WSNs. In Section 7.1.2, we provide a background review on information content valuation techniques. Following this, we discuss existing adaptive routing algorithms in Section 7.1.3. Then, we focus on efficient energy management schemes for WSNs in Section 7.1.4.

7.1.1 Data Sampling

In this section, we focus on data sampling algorithms within the WSN domain. Here, it is typically insufficient to have sensors deployed with a fixed sampling rate. In particular, due to the limited energy capacity of each individual sensor, it is crucial to avoid sampling unnecessary data (e.g. data that does not contain any new information). Since different environments provide different characteristics, the sensors need to learn an efficient sampling rate, that fits their surroundings, in order to avoid sampling unnecessary data, and thus, to improve their performance in information collection. As a result, it is necessary to use some form of adaptive sampling approach on each sensor in the network. Generally speaking, adaptive sampling is often described as "intelligent sampling" (Guestrin *et al.*, 2005; Krause *et al.*, 2006), since it is adaptive to the unknown environmental characteristics. In particular, an adaptive sampling algorithm is here defined as a protocol (i.e. a set of policies) that is responsible for adaptively setting the sampling rate (i.e. *how often* a node is required to sample during a particular time interval) and the schedule (i.e. *when* a node is required to sample) of each of the individual nodes in a network.

Existing algorithms can be classified as to whether they use temporal or spatial correlations (or both) in order to make effective sampling decisions. With respect to spatial

correlations between sensors, the challenge of calculating informative locations has been thoroughly studied by Guestrin *et al.* (2005). In this approach, the spatial correlations within the monitored environment are assumed to be known. These correlations are modelled by using a multi–variate Gaussian and are learnt during the initial deployment of the network. Based on this information, an informative subset of the sensors is then selected to provide information to a base station (i.e. BS), while the rest of the nodes are removed in order to reduce cost. Krause *et al.* (2006) extend this work by taking communication cost into account, making an explicit trade–off between the energy consumption of sampling and communication of each sensor. Finally, Willett *et al.* (2004) have studied the backcasting adaptive sampling method in which multiple nodes that are spatially correlated form small subsets of nodes that then communicate their information to a local data aggregation coordinator. Based upon this information, the coordinator then selectively activates additional nodes (by instructing them to take samples) in order to reduce uncertainty below a specified target level. While the first two techniques are decentralised, the third method uses a centralised coordination mechanism, that contains all the drawbacks of the centralised regime (as discussed in Section 1.2).

To handle temporal correlations, the *utility based sensing and communication* (USAC) algorithm was proposed by Padhy *et al.* (2010). This is a decentralised control protocol for adaptive sampling, designed for an environmental WSN, known as Glacsweb, intended to measure subglacial movement (Martinez *et al.*, 2004). In this approach, temporal variations in the environmental parameter being sensed are modelled as a piece–wise linear function, and then the algorithm uses a pre–specified confidence interval parameter in order to make real–time decisions regarding the sampling rate of the sensor nodes. Moreover, linear regression is used to predict the value of future measurements, and if the actual sensor reading exceeds the confidence interval parameter, the sensor starts sampling at an increased rate. However, since the algorithm does not explicitly perform any forward planning, the sensor can rapidly deplete its battery if the increased sampling rate is constantly re–triggered by data that is far from linear.

Furthermore, in ab application where sensor networks are tasked to monitor tidal sea level, Kho *et al.* (2009) proposed a decentralised algorithm using an information metric that represents the temporal variation in the environmental parameter being sensed. This algorithm, in contrast to USAC, takes energy harvesting into account, and thus, enables the sensors to make long–term plans. In particular, this algorithm aims to maximise the information that a sensor collects over a particular time interval subject to energy constraints, and this involves planning exactly when, within the specified time interval, to take a constrained number of samples. The algorithm takes the information provided by the information metric into account when creating a sampling schedule at the beginning of each day. This algorithm also considers the amount of residual energy

left in the sensor's battery, and the amount of information that can be collected at different times of the day, based on past experience.

In addition, Jain and Chang (2004) used a similar prediction technique to set the sampling rate adaptively. Their approach employs a *Kalman filter* (KF) based estimation technique wherein the sensor can use the KF estimation error to adaptively adjust its sampling rate within a given range, autonomously. When the desired sampling rate violates the range, a new sampling rate is requested from a control server. The server allocates new sampling rates under the constraint of available resources such that the KF estimation error over all the active streaming sensors is minimised. However, the main drawback of this technique is that it is centralised, and thus, it is not feasible to operate it in a decentralised setting (see the requirements of our research in Section 1.1).

Finally, the algorithm proposed by Osborne *et al.* (2008) uses a multi–output *Gaussian process* (GP) to explicitly model both temporal and spatial correlations between a small number of sensors. The GP is used for adaptive sampling whereby it can determine both the time, and the sensor from which the next sample should be taken, to ensure that the uncertainty regarding the environmental parameter being measured at each sensor location stays below a pre–specified threshold. However, the algorithm is centralised, since it requires information from all of the sensors in order to model the spatial correlations between them, and it is relatively computationally expensive.

In this chapter, we assume that our sampling protocol is a generic adaptive sampling protocol. We have only one restriction; that is, the algorithm should be decentralised. Each agent should be able to autonomously and independently set its own sampling rate and schedule (for more details see Section 7.2). Given this, decentralised techniques, such as the sampling protocol of USAC, or the algorithms proposed by Guestrin *et al.* (2005), Krause *et al.* (2006), and Kho *et al.* (2009), can be used here. On the other hand, due to their centralised manner, the algorithms proposed by Willett *et al.* (2004), Jain and Chang (2004) and Osborne *et al.* (2008) are not suitable for our model.

7.1.2 Information Content Valuation

In order to distinguish important and unimportant data from each other, and thus, to achieve a more efficient information collection in terms of maximising the total information delilvered to the BS, an efficient information metric is required to determine the information content of the collected or transmitted data. In our case, this metric is provided by an *information content valuation function*.

Within the tracking literature, where spatially correlated sensor readings typically represent the estimated position of a target, there are a number of standard techniques for defining this function. Most of the works use *Fisher information*, whereby the estimated

position of the target is represented as a multidimensional probability distribution, and Fisher information is used to quantify the uncertainty represented by this distribution (Bar-Shalom *et al.*, 2001; Chu *et al.*, 2002; Frieden, 2004; Zhao and Guibasn, 2004). For example, Chu *et al.* (2002) used acoustic sensors to localise a target. To quantify the information gain of the measured data provided by each sensor, they used the Fisher information matrix as follows. Let $\mathbf{x} = \{\mathbf{x_1}, \mathbf{x_2}, \dots\}$ denote the set of unknown parameters of the target, and $\mathbf{z} = \{z_1, z_2, \dots, z_N\} \in \mathbb{R}^N$ denote the set of sensor measurements, where N is the number of the sensors. Thus, the ij^{th} component of the Fisher information matrix is:

$$\mathbf{F}_{ij}(\mathbf{x}) = \int_{\mathbb{R}^N} p(\mathbf{z}|\mathbf{x}) \frac{\partial}{\partial x_i} \ln p(\mathbf{z}|\mathbf{x}) \frac{\partial}{\partial x_j} \ln p(\mathbf{z}|\mathbf{x}) \; d\mathbf{z}.$$

Other approaches have used mutual information as a criteria for sampling. For example, Krause *et al.* (2006) modeled the spatial correlations of locations, in order to determine efficient sensor placements, whereby a maximal information value can be collected by data sampling. In their model, V denotes the set of possible locations, A denotes the set of observable locations and s is an unobservable location. Let $\mathbf{X}_A$ denote the set of observable random variables associated with the locations A, and X_s be the random variable associated with location s. In order to make predictions at a location S (i.e. to calculate conditional distributions $p(X_s = x_s|\mathbf{X}_A = \mathbf{x}_A)$), they used the following conditional entropy:

$$H(X_s|\mathbf{X}_A) = -\int_{x_s,\mathbf{x}_A} p(x_s, \mathbf{x}_A) \log p(x_s|\mathbf{x}_A) \; dx_s \; d\mathbf{x}_A.$$

Intuitively, this quantity expresses how "peaked" the conditional distribution of X_s is, given X_A is around the most likely value, averaging over all possible observations $X_A = x_A$ the sensors can make. To quantify how informative the set of data collected from locations A is, they used the criterion of *mutual information* (MI):

$$F(A) = I\left(X_A, X_{V/A}\right) = H\left(X_{V/A}\right) - H\left(X_{V/A}X_A\right).$$

This criterion expresses the expected reduction of entropy of all locations V/A where sensors were not placed, after taking into account the measurements of sensors at locations in set A.

Similarly, Osborne *et al.* (2008) used a Gaussian process (GP) to model both the spatial and temporal correlations and delays between nodes, using Bayesian Monte Carlo techniques to marginalise over the unknown hyper–parameters that describe the correlations and delays. They then use the variance of the GP's predictions in order to perform active data selection, which is a decision problem concerning which observations should be taken, deciding when and where to take samples to maintain this variance below

a prescribed target level. Their algorithm is computationally efficient as the samples are learned from the data in an online fashion, and thus, it is capable of performing real–time information processing.

Several other techniques for valuing information include *Shannon entropy*, which is mainly used in signal compression (or coding), target tracking, and information fusion techniques in WSNs (Cover and Thomas, 2006; Hwang *et al.*, 2004). For instance, Hwang *et al.* (2004) used a belief vector to probabilistically represent the identity of a target. In their work, they considered the problem of combining two belief vectors of the same target from two different sensors (i.e. data fusion). Here, information fusion can be formulated as an optimisation problem such that the fused information is the one that minimises a cost function which represents a performance criterion. This cost function is modelled by the Shannon entropy. More precisely, let b_1 and b_2 denote the belief vectors before data fusion, and $b^{'}$ denote the information value of the fused packet. Furthermore, let

$$b^{'} = wb_1 + (1 - w)\, b_2$$

be the fusion strategy. The goal is to determine w such that it minimises the Shannon entropy defined as:

$$H\left(b^{'}\right) = -\sum_{i=1}^{n} b^{'}(i) \log b^{'}(i),$$

where $b^{'}(i)$ denote the probability that the target is in belief state i. Informally, Shannon entropy characterises the average amount of information which is gained from a certain set of events. The entropy is maximal when all the events' outcomes are equally likely and, therefore, we are uncertain which event is going to happen. When one of the events has a much higher chance of happening than the others, then the uncertainty (or entropy) decreases. Information value can thus be quantified as the difference between the probabilities of the random event.

In addition, Padhy *et al.* (2010) use the *Kullback–Leibler* (KL) divergence to model the information value of collected data in USAC. Here, KL divergence is a measure of the information gain between a prior and a posterior probability distribution (i.e. the distribution over possible measurements before and after a new item of data has been received). The larger this measure, the less the previous model was capable of explaining the value of the new data, and thus, the more it has to be updated. More recently, Kho *et al.* (2009) use the mean Fisher information over a period as a measure that is proportional to the value of information. In this thesis, we assume that our model is capable of using any of these techniques, and thus, we do not specify any restrictions on the information valuation technique in use (for more details see Section 7.2).

7.1.3 Information–Centric Routing

Routing is the process of delivering a message from a source node to a BS inside the same network. A routing algorithm determines actions that a node can use to forward data towards the BS, namely (i) transmitting (i.e. which data packets the node should choose to transmit, and to which node), and (ii) receiving (i.e. how many packets a node is required to receive from the other nodes during a transmission period). Since routing is responsible for transporting the data collected by the network to the BS, the efficiency of the routing algorithm significantly affects the overall performance of the network. In fact, routing is one of the most studied areas within the WSN domain, and thus, a large number of algorithms have been proposed for adaptive and efficient data routing in WSNs from many different perspectives. These include, but are not limited to, algorithms that address: energy efficiency, delay sensitiveness, security, and reliability (Ahdi *et al.*, 2007; Akkaya and Younis, 2005; Al-Karaki and Kamal, 2004; Singh *et al.*, 1998). However, these algorithms typically do not distinguish important packets from unimportant ones. Thus, this may lead to inefficient performance in terms of information collection, since it may occur that less important data is delivered to the BS, while the more important packets are not forwarded at all. Given this, in this section, we focus on routing approaches that are information–centric (Braginsky and Estrin, 2002; Merrett, 2008). In particular, these approaches aim to maximise the total information value delivered to the BS. In so doing, they typically use information content valuation techniques in order to determine more important data packets.

One of these algorithms, *directed diffusion* (DD), has been developed by Intanagonwiwat *et al.* (2003). In DD, the BS sends out a data collection query description by flooding the query to the entire network. That is, data collection happens only when the BS needs a certain type of data. However, since data collection applications (e.g environmental monitoring or area surveillance) typically require continuous data delivery to the BS, a significant number of queries will be sent to the network. In this case, the communication cost of DD caused by query floodings is high, meaning DD is not suitable for long–term information collection. To avoid flooding, the *rumor routing* (RR) protocol routes the queries to the nodes that have observed a particular event to retrieve information about the occurrence of the event, and thus, it reduces the total communication cost (Braginsky and Estrin, 2002). However, rumor routing performs well only when the number of events is small. For a large number of events, the algorithm becomes infeasible due to the increase in the cost of maintaining node–event tables in each node.

Apart from the aforementioned approaches, in which information is collected by sending explicit queries from the BS, other methods focus on continuous information collection. That is, they provide information collection, without the need of sending any queries, during the whole operation of the network. For instance, USAC (see Section 7.1.1), considers the remaining battery power of the communicating nodes and the importance

of the data being transmitted, in order to determine the appropriate routing path for the packet. In a similar vein, the *adaptive routing algorithm* (ARA), that has been developed by Zhou and de Roure (2007), in addition to the battery level and the importance of data, takes the link cost (assumed to be proportional to the distance) between the nodes into account when routing packets. However, these protocols are not designed for solving the maximal information throughput routing problem, but rather to identify optimal paths between each node and the BS, that can be used for forwarding data (see Section 7.2.4 for more detail).

Finally, an interesting last class of information–centric routing protocols are those that use a market–based control (MBC) paradigm. The use of MBC in WSN allows the use of tools from general equilibrium theory to analyse the behaviour and correctness of a decentralised system. The main market–based protocol includes *self organised routing* (SOR), proposed by Rogers *et al.* (2005), and *self organising resource allocation* (SORA), proposed by Mainland *et al.* (2005). In more detail, SOR is a mechanism–design based distributed protocol that aims to maximise the network's lifetime. Each node is designed to follow locally selfish strategies which, in turn, result in the self organisation of a routing network with desirable global properties. The protocol consists of a communication protocol, equipping nodes with the ability to find and select a node that is willing to act as a mediator for data relaying, and a payment scheme, whereby a node is rewarded for forwarding messages to the destination. Specifically, the communication scheme identifies potential mediators, the payment scheme allows the sensors to make local selfish decisions which result in good system–wide performance. In contrast, SORA defines a virtual market in which nodes sell goods (e.g. data sampling, data relaying, data listening, or data aggregation) in response to global price information that is established by the end–user. However, this approach again involves an external coordinator to determine the price and it is not clear how this price determination should actually be done in practice. In sum, although these algorithms are based on the multi–agent systems approach, which is clearly related to our model, we do not follow their perspective. In contrast, within this chapter, we concentrate on networks where sensors maximally cooperate with each other.

7.1.4 Energy Management

An energy management policy is responsible for allocating energy budgets to sensory tasks, such as sampling and routing data. Most of these policies, however, are typically integrated into the routing algorithm, ignoring the task of sampling. In particular, these methods assume that the data are already sampled, and thus, they focus only on delivering data towards the BS. Furthermore, they typically follow the concept of energy–awareness; that is, they aim to minimise the energy consumption of each node, while data forwarding is still to be done.

Given this, a number of energy–aware algorithms use clustering techniques to minimise energy consumption in sensor networks through the rotation of cluster–heads such that the high energy consumption in communicating with the BS is spread across all nodes. These algorithms include *low energy adaptive clustering hierarchy* (LEACH), proposed by Heinzelman *et al.* (2000), and *power efficient gathering in sensor information systems* (PEGASIS), proposed by Lindsey and Raghavendra (2002). In general, these methods make good effort on minimising the energy consumption by electing cluster–heads, each of which is responsible for relaying the data from a subset of nodes back to the BS in an intelligent way. These cluster–heads all need to be placed inside the BS's radio range as they communicate with it directly. Thus, this assumption limits the size of the monitoring environment, since the wireless radio range of the BS is limited. Moreover, these single cluster–heads can become a communication bottleneck of the network, since in each round the cluster–heads need to communicate with a large number of nodes within their cluster. Hence, this aspect contains some of the drawbacks of the centralised control regime.

The life span of the network can also be lengthened by reducing the total energy consumption needed to deliver the packets to the BS. From this perspective, Dekorsy *et al.* (2007) proposed an approach that jointly controls the routing and energy management, in order to achieve efficient data forwarding. In particular, their approach aims to minimise the total energy consumption of each node, while the collected data has to be delivered to the BS using multipath routing (i.e. there can be multiple routing paths between a node and the BS). In so doing, the approach considers each node's residual energy level, the transmission power level, and maximal communication bandwidth. This approach, however, assumes that the data is already sampled, and that future data is not taken into consideration when optimal routing paths are calculated. This implies that this approach is designed for single–shot optimisation (i.e. it only considers one time step), rather than long–term performance maximisation, in which more than one time step is taken into account. For long–term optimisation, it has to recalculate the optimal paths at each time step, and this requires significant computational resources.

In addition, another way to to lengthen the life span of the network is to perform *energy balancing* (Dinga *et al.*, 2004). That is, to maximise the residual energy level of the bottleneck node (i.e. the node with the least energy level) in the network during the routing. In this vein, Ok *et al.* (2009) used a metric to take the energy cost of transmission, as well as the sensors' remaining energies into account. This metric gives rise to the design of the *distributed energy balanced routing* (DEBR) algorithm, to balance the data traffic of sensor networks in a decentralised manner. Furthermore, Li *et al.* (2007) proposed a global–energy balancing routing scheme (GEBR) for real–time traffic. Now, while both of these algorithms perform well in prolonging the lifetime of the WSN, similar to the

approach of Dekorsy *et al.* (2007), they assume that the data has already been sampled. Thus, they are not designed for long–term information collection.

More recently, Merrett (2008) developed the *information managed energy aware algorithm for sensor networks* (IDEALS) protocol, which aims to extend the network lifetime of WSNs. IDEALS is an application specific heuristic protocol as it requires that every sensor node decides its individual network involvement based on its own energy state and the importance of information contained in each message. In particular, IDEALS groups the packets into levels of packet priority (PP), according to their importance. It also maintains a set of different energy levels, most likely in simulation, for a particular sensor node, which it classifies as energy priority levels (EP). Now, if the EP of a particular sensor node is higher than the PP level of a packet within the sensor's memory, the sensor will not forward that packet. This results in a trade–off between sending important data and balancing the energy consumption of the network. However, since the EP levels have to be set *a priori* before the deployment of the network, it is necessary to finely tune these levels in order to achieve a good performance in different environments. Thus, IDEALS fails to fulfil Requirement 4 (adaptivity).

Finally, similar to IDEALS, USAC uses the *opportunity cost* of the energy used by each sensor to balance the energy consumption of the tasks of sampling and forwarding. That is, by evaluating its own opportunity cost, each sensor can decide whether it should spend energy on sampling or forwarding, depending on which is the more preferable opportunity for the sensor. Moreover, USAC also considers the total energy consumption required to transmit a packet along a particular path as well. This method, since it can vary the energy budgets allocated to the sensory tasks, is most related to our work. As a result, we will compare our approach against the performance of USAC within our empirical evaluations.

7.2 System Models and Problem Definitions

Having described the literature of relevance in the previous section, we now introduce a formalisation of the long–term information collection problem for WSNs. To this end, we first provide a formal description of the WSN system in Section 7.2.1. In particular, we describe the models of adaptive sampling, information content valuation, data routing, and energy management policies that play fundamental roles in efficient information collection of WSNs. Here, we also discuss the assumptions, on which the model formalisation is based. Following this, in Section 7.2.2, we formulate the main objective of our research: that is, to achieve efficient long-term information collection in WSNs. Finally, we decompose the information collection problem into the two separate sub–problems

described in Section 1.3: (i) energy management; and (ii) maximal information throughput routing, which we introduce in Sections 7.2.3, and 7.2.4, respectively.

7.2.1 The Wireless Sensor Network Model

In order to formalise the long–term information collection challenge introduced earlier, we first need to introduce a suitable WSN model. Given this, we now present our WSN model, that covers the energy management, sampling, information content valuation, and routing components, respectively.

Recall that we here pursue a decentralised control model. This, however, implies that in order to achieve system–wide goals, the nodes must typically coordinate their actions with their neighbours (e.g. to forward data or to track objects). In addition, we also require that the nodes must be able to autonomously adapt their behaviour, without having global information about the system. Such requirements naturally lend themselves to a *multi–agent system* (MAS) perspective (Lesser *et al.*, 2003; Pechoucek and Marik, 2008; Soh and Tsatsoulis, 2005), in which each sensor is represented by an agent, which autonomously and cooperatively acts, in order to achieve system-wide objectives (Jennings, 2001). As a result, we also pursue a multi–agent system model, whereby sensor nodes are represented as agents.

Now, since our main focus is on the control side of the WSN, we make the following assumptions about the physical world of the network, in order to simplify the complexity of the model:

- The network that we are studying is not a mobile network (i.e. the agents cannot change their location), however, link failures, node failures and node additions are taken into account. That is, the network can be *topologically dynamic, but not mobile.*

- In our model, the *energy consumption of memory management* (i.e. reading from memory and writing to memory) *is negligible* compared to the energy consumption of data sampling and forwarding. This assumption is reasonable according to the experimental studies reported in Mathur *et al.* (2006) and Anastasi *et al.* (2004).

- We also assume that once the communication channel is set between two nodes, data transmission between these nodes is *perfect* (i.e. no data loss occurs). This assumption is reasonable, especially in networks where there is a demand of high quality of service (QoS) (Younis *et al.*, 2004). In particular, if the ratio of successful transmission of a communication channel is low (i.e. the QoS is low), then that communication channel cannot be established. In order to guarantee high

QoS within WSNs, efficient techniques can be used, such as time synchronisation policies (Degesys and Nagpal, 2008; Elson and Estrin, 2001; Sundararaman *et al.*, 2005), or medium access control (MAC) protocols that control the data transmission of each node (Demirkol *et al.*, 2006; Wu and Biswas, 2007). By using the aforementioned techniques, we can guarantee that no data loss occurs during data transmission.

Given this, we can formulate the WSN model as follows. Let $I = 1, 2, \ldots, N$ be the set of agents in the network, which contains one base station, denoted BS[1]. We assume that each agent knows its distance in hops from the BS. This can be achieved by using any of the standard shortest path algorithms (e.g. distributed breadth-first search or distributed Bellman–Ford). Furthermore, each agent can only communicate with those that are inside its communication range, and different agents may have different ranges. For the sake of simplicity, we split the time line into steps. That is, hereafter we assume that time is discrete, and can be denoted with the sequence of $t = 0, 1, 2, \ldots$.

We consider three specific kinds of energy consumption for each agent in the network, namely: the energy required to (i) acquire (i.e. sample); (ii) receive; and (iii) transmit a single data packet (we assume that each packet has the same size in bytes). Given this, let e_i^{S}, e_i^{Rx}, and e_i^{Tx} denote the energy consumption that agent i has to spend for sampling, receiving, and transmitting a single data packet, respectively. We only consider the aforementioned energy consumptions, and we disregard the energy required for other types of processing since it is negligible in comparison (Mathur *et al.*, 2006; Merrett, 2008).

Let B_i denote the initial battery capacity, and let $B_i(t)$ denote the residual battery capacity of agent i at time step t, respectively. Note that $B_i(1) = B_i$. At each time step t, the energy consumption of agent i cannot exceed $B_i(t)$ in our settings. In addition, since the length of a time step is finite, and the physical time needed to execute a sensory action is non–zero, there is a threshold on the maximal number of packets an agent can sample, transmit, or receive (Anastasi *et al.*, 2004; Mathur *et al.*, 2006). As a result, let N_i^{S}, N_i^{Rx}, and N_i^{Tx} denote the maximal number of packets that agent i can sample, receive, and transmit within a time step, respectively.

For data sampling, since our goal is not to develop new sampling techniques, we use existing sampling techniques from the literature. Specifically, we focus on *adaptive data sampling* techniques. Such policies have been advocated as the way to achieve accurate estimates of the environmental conditions, whilst minimising redundant sampling of the environment. Relevant examples can be found in Section 7.1.1. To calculate the importance of sampled data, we use information content valuation methods. Similar to the sampling case, any existing technique from the literature can be used for this (see

[1] Our model can easily be extended to cover systems with multiple base stations.

Section 7.1.2 for more details). Furthermore, we also assume that the information value of the collected data is *discounted* over time by a durability factor $\lambda \in (0, 1]$ (i.e. it loses value as time passes by), if it is not delivered to the BS yet. This assumption is justified by the fact that in many applications, more up to date information is preferable to older information. Since our main focus is on networks without real–time delivery constraints (see Section 1.2 for more details), we assume that the information durability factor is typically high (i.e. $\lambda > 0.5$). The intuition of this assumption is that with a higher information durability factor, the collected information can then be delayed for a longer time, without losing much of its value, before it is delivered to the BS. Note that within our model, the information value of non–collected data (i.e. data that are not sampled yet by the agents) may also decay over time. However, we assume that the underlying sampling method can efficiently sample data so that important data can be collected earlier than less important data.

In existing routing protocols, agents typically forward data to other agents, which are closer to the BS, either in terms of physical distance or number of hops. Thus, following this concept, we assume that in our model, agents can send data to those which are closer to the BS in terms of number of hops. Finally, we assume that data sampled or received at each agent i at step t can only be forwarded from step $(t + 1)$. This assumption is also reasonable, since without it, newly sampled data could be delivered to the BS instantaneously.

7.2.2 The Long–Term Information Collection Problem

Given the model that considers adaptive sampling, routing, information valuation and energy management of WSNs, we now give a formal description of the research objective. That is, to maximise the total collected information in WSNs, in a given finite time interval. In more detail, let $\mathbf{S}_i(t)$, $\mathbf{Rx}_i(t)$ and $\mathbf{Tx}_i(t)$ denote the set of sampled, received and transmitted data packets of agent i at time step t. Let p denote a single data packet, whose information value at time step t is $v(p, t)$. Furthermore, we assume that the WSN operates in the finite time interval $[0, T]$. Given this, our objective is to maximise the total information value delivered to the BS over the time interval $[0, T]$, which can be formulated as follows:

$$\max \sum_{t=0}^{T} \left\{ \sum_{p \in \mathbf{Rx}_{BS}(t)} v(p, t) \right\}. \tag{7.1}$$

Here, $\mathbf{Rx}_{BS}(t)$ denotes the set of packets that the BS receives at time step t. We have to take the following constraints into account:

$$\mathbf{Tx}_i(t) \subseteq \mathbf{Q}_i(t) \tag{7.2}$$

for each agent i and time step t, where $\mathbf{Q}_i(t)$ is the set of total transmittable data packets in the memory. That is, the set of transmitted data is the subset of the total

data that are ready to be transmitted (packets that were sampled or arrived until the previous time step) of each agent i. Furthermore,

$$\mathbf{Q}_i(t+1) = (\mathbf{Q}_i(t) / \mathbf{Tx}_i(t)) \cup \mathbf{S}_i(t) \cup \mathbf{Rx}_i(t) \tag{7.3}$$

for each agent i. Note that $\mathbf{Q}_i(t) / \mathbf{Tx}_i(t)$ denotes the set of packets that is in $\mathbf{Q}_i(t)$ but not in $\mathbf{Tx}_i(t)$ (i.e. exclusion). That is, the set of transmittable data of agent i at time step $(t+1)$ is the union of the sets of residual data (i.e. $(\mathbf{Q}_i(t) / \mathbf{Tx}_i(t))$), the received data and the sampled data at time step t. Taking the energy constraints into account, we have the following:

$$\begin{aligned} |\mathbf{S}_i(t)| &\leq N_i^{\mathrm{S}}, \\ |\mathbf{Rx}_i(t)| &\leq N_i^{\mathrm{Rx}}, \\ |\mathbf{Tx}_i(t)| &\leq N_i^{\mathrm{Tx}} \end{aligned} \tag{7.4}$$

for each agent i, where $|\{.\}|$ denotes the size of set $\{.\}$.

Furthermore, for each $p \in \mathbf{S}_i(k) \cup \mathbf{Rx}_i(t)$ (i.e. received data or sampled data of agent i at time step t), that is not delivered to the BS before time step t:

$$v(p, t+1) = \lambda v(p, t), \tag{7.5}$$

where $\lambda \in (0, 1]$ is the durability coefficient. That is, the information value of packet p is decayed with the durability factor λ, as time goes by.

As mentioned in Section 1.3, to efficiently solve the problem formulated in Equation 7.1, we separate the study of the energy management and routing of the WSN, whilst we assume that efficient sampling and information content valuation can be achieved by using existing techniques. Given this, Section 7.2.3 discusses the energy management problem in more detail, whilst Section 7.2.4 focuses on the routing problem.

7.2.3 The Energy Management Problem

As mentioned in Section 1.3, the definition of the energy management problem is based on the observation that since each agent can sample, receive or transmit data, it is necessary for the agents to vary the energy budget they associate with each of these action types, so that their overall performance can effectively adapt to environmental changes. That is, by adaptively setting the value of the energy budgets assigned to the sensory tasks, the agents can decide whether to put more effort on sampling (e.g. when significant events are occurring in the monitored area), receiving important data from the others (e.g. when they have collected high value information that has to be delivered to the BS), or transmitting data (e.g. when the delivery of data cannot be delayed too long). With such capabilities, our hypothesis is that the agents should achieve better

performance than systems without the ability to adapt in this fashion. However, the agent here has to deal with the problem of exploration *versus* exploration as follows. In order to find the optimal combination of budget allocation (exploitation), the agents first have to learn the efficiency of each combination (exploration). As a result, if the agent only focuses on learning the optimal combination, the total collected information of that agent over the operation time might not be maximal, since the agents has to try out all the combinations (including those with low efficiency). On the other hand, if the agent decides to focus on the best combination so far, it may miss the chance to find a better combination that results in better overall performance (i.e. better collected information over a long term).

Consequently, the energy management problem, that we are faced with, is a sequential decision making problem where at each time step t, each agent i has to choose a combination of energy budget allocations for sampling, receiving, and transmitting, respectively. Following this, agent i evaluates the efficiency of the chosen combination by measuring the amount of sampled, received, and transmitted information within that time step, with respect to the chosen energy budgets. The goal of each agent i is to find a sequence of decisions (i.e. learning method) that efficiently tackles the trade–off between exploration and exploitation, and the dynamic behaviour of the environment, leading the overall system to achieve maximal long–term information collection.

More precisely, let $B_i^{\mathrm{S}}(t)$, $B_i^{\mathrm{Rx}}(t)$, and $B_i^{\mathrm{Tx}}(t)$ denote the energy budgets that agent i allocates to sampling, receiving and transmitting at time step t, respectively. That is, at each time step, agent I makes a decision of choosing values for $B_i^{\mathrm{S}}(t)$, $B_i^{\mathrm{Rx}}(t)$, and $B_i^{\mathrm{Tx}}(t)$. In so doing, beside the constraints given in Section 7.2.2, it has to take into account the following:

$$\begin{aligned} e_i^{\mathrm{S}} \left|\mathbf{S}_i(t)\right| &\leq B_i^{\mathrm{S}}(t), \\ e_i^{\mathrm{Rx}} \left|\mathbf{Rx}_i(t)\right| &\leq B_i^{\mathrm{Rx}}(t), \\ e_i^{\mathrm{Tx}} \left|\mathbf{Tx}_i(t)\right| &\leq B_i^{\mathrm{Tx}}(t). \end{aligned} \tag{7.6}$$

These constraints demonstrate that the energy consumption of each action made by agent i cannot exceed the residual energy budget of each task (Equation 7.6), and the action thresholds (Equation 7.4) given in time step t. Furthermore, we have:

$$B_i^{\mathrm{S}}(t) + B_i^{\mathrm{Rx}}(t) + B_i^{\mathrm{Tx}}(t) \leq B_i(t). \tag{7.7}$$

This constraint demonstrates that the total energy consumption of the actions taken by agent i cannot exceed the energy budget given in time step t. The residual energy budget of the next time step then can be calculated as:

$$B_i(t+1) = B_i(t) - \left(B_i^{\mathrm{S}}(t) + B_i^{\mathrm{Rx}}(t) + B_i^{\mathrm{Tx}}(t)\right). \tag{7.8}$$

That is, we assume that the total amount of the allocated energy has to be used within each time step. This assumption is reasonable, since in real–world WSNs, energy budget

allocation in fact means that the node turns on the corresponding sensory module for a certain time interval. Within this interval, the module consumes energy. By setting the length of this interval (in which the module is turned on), the node can set the size of the energy budget. Given all this, the energy management problem can be formalised as follows:

Definition 7.1. Within the *energy management problem*, each agent i has to sequentially choose a sequence of 3–tuples $\langle B_i^{\mathrm{S}}(t), B_i^{\mathrm{Rx}}(t), B_i^{\mathrm{Tx}}(t)\rangle$ at time step t in order to maximise the objective given in Equation 7.1. A tuple $\langle B_i^{\mathrm{S}}(t), B_i^{\mathrm{Rx}}(t), B_i^{\mathrm{Tx}}(t)\rangle$ represents the energy budgets allocated to data sampling, receiving, and transmission, respectively. Each of these 3–tuples has to satisfy the following constraints:

- The number of sampled, received, and transmitted packets cannot exceed the allocated budgets (see Equation 7.6).
- The total energy budget allocation cannot exceed the residual energy budget (see Equation 7.7).
- The next residual energy budget is the difference between the previous residual energy budget and the allocated energy budgets within the previous time step (see Equation 7.8).

In Section 7.3, we propose a budget–limited MAB learning approach, in order to efficiently tackle this problem.

7.2.4 The Maximal Information Throughput Routing Problem

Having described the energy management problem, we now discuss the maximal information throughput routing problem, which aims to maximise the total information that can be forwarded between neighbouring layers (i.e. the group of agents that are the same distance from the BS) of agents. Given this, we group the agents within the network into layers, such that $\mathfrak{L}_l$ denotes the set of agents that are l hops from the BS. Let L denote the number of layers in the network. Note that the BS itself is layer 0. Thus, we have the following:

Definition 7.2. The *maximal information throughput problem* is the optimisation problem where agents in layer $\mathfrak{L}_l$ have to perform the maximal total information throughput to layer $\mathfrak{L}_{l-1}$ in time step t, with respect to the energy budgets of each agent.

The formulation of the problem can be described as follows:

$$\max\left\{\sum_{i\in\mathfrak{L}_l}\sum_{p\in Tx_i(k)} v(p,t)\right\}, \tag{7.9}$$

with respect to the following constraints:

$$E_i^{\mathrm{Tx}} \left|\mathbf{Tx}_i\left(t\right)\right| \leq B_i^{\mathrm{Tx}}\left(t\right) \tag{7.10}$$

for each $i \in \mathfrak{L}_l$, where $\mathbf{Tx}_i\left(t\right)$ is the set of transmitted data of node i at time step t, and $v\left(p,t\right)$ is the information value of packet p at t. That is, each sender agent cannot exceed its transmitting energy budget during its data transmission operation. Furthermore,

$$E_j^{\mathrm{Rx}} \left|\mathbf{Rx}_j\left(t\right)\right| \leq B_j^{\mathrm{Rx}}\left(t\right) \tag{7.11}$$

for each $j \in \mathfrak{L}_{(l-1)}$, where $\mathbf{Rx}_j\left(t\right)$ is the set of received data of node i at time step t. Thus, each receiver agent cannot exceed its receiving budget during data receiving. Finally, constraints described in Equations 7.2, 7.3, and 7.5, that express the conservation of information within our setting, have to be taken into account as well.

In order to solve this problem, we propose two decentralised algorithms, one is optimal, but with significant communication costs, whilst the other is near–optimal, but with reduced costs. We describe these algorithms in more details in Section 7.4. Moreover, we will show that the proposed algorithm, in conjunction with the budget–limited MAB algorithms described in the previous three chapters, outperform information collecting state–of–the–art algorithms in WSNs.

7.3 Multi–Armed Bandit Based Energy Management

Given the problem definitions described above, we now concentrate on the energy management problem presented in Definition 7.1. Therefore, we first describe the MAB learning based energy management approach in Section 7.3.1. Then we analyse the computational complexity of this approach in Section 7.3.2. In particular, we show that our approach has linear running time, and linear memory usage, compared to the number of each agent's available options of energy budget allocation.

7.3.1 Using Multi–Armed Bandits for Energy Management

Within this section, we show how to apply the budget–limited MAB model to the energy management problem described in Section 7.2.3. In so doing, consider the formal model we introduced in Section 7.2. Recall that within this model, each agent i has a residual energy budget $B_i\left(t\right)$ for each time slot t, such that $B_i\left(1\right) = B_i$ is the initial battery capacity of agent i. Furthermore, agent i has to allocate budgets $B_i^{\mathrm{S}}\left(t\right)$, $B_i^{\mathrm{Rx}}\left(t\right)$, and $B_i^{\mathrm{Tx}}\left(t\right)$ to sampling, receiving and transmitting, respectively. The energy budget allocation, however, has to satisfy Equation 7.7.

Given this, we can formulate the energy management problem of a single agent as a budget–limited MAB as follows. We first define the set of arms, the pulling cost of each arm, and the budget of the agents. Then we determine the reward function of each action. The latter is the mechanism that assigns reward values to the action of the agent at each time slot.

In so doing, let us consider a decision given in Definition 7.1 that agent i makes at time slot t. Recall that the number of packets that agent i can sample, receive, and transmit is limited (see Equation 7.4). Thus, we assume that the following holds:

$$\begin{aligned} B_i^{\mathrm{S}}(t) &\leq e_i^{\mathrm{S}} N_i^{\mathrm{S}}, \\ B_i^{\mathrm{Rx}}(t) &\leq e_i^{\mathrm{Rx}} N_i^{\mathrm{Rx}}, \\ B_i^{\mathrm{Tx}}(t) &\leq e_i^{\mathrm{Tx}} N_i^{\mathrm{Tx}}. \end{aligned} \tag{7.12}$$

This assumption is reasonable, since it indicates that since the number of sampled, received, and transmitted packets are all limited due to physical constraints (see Section 7.2.2 for more detail), it is inefficient to allocate more energy than that the physical constraints allow. As a result, for each agent i, consider the following set of 3–tuples:

$$\mathfrak{A}_i := \left\{ \left\langle n_i^{\mathrm{S}} e_i^{\mathrm{S}}, n_i^{\mathrm{Rx}} e_i^{\mathrm{Rx}}, n_i^{\mathrm{Tx}} e_i^{\mathrm{Tx}} \right\rangle \right\}, \tag{7.13}$$

with respect to:

$$\begin{aligned} 0 &\leq n_i^{\mathrm{S}} \leq N_i^{\mathrm{S}}, \\ 0 &\leq n_i^{\mathrm{Rx}} \leq N_i^{\mathrm{Rx}}, \\ 0 &\leq n_i^{\mathrm{Tx}} \leq N_i^{\mathrm{Tx}}, \end{aligned} \tag{7.14}$$

$$0 < n_i^{\mathrm{S}} + n_i^{\mathrm{Rx}} + n_i^{\mathrm{Tx}}. \tag{7.15}$$

The last inequality guarantees that the tuple $\langle 0, 0, 0 \rangle$ is excluded from the set (i.e. the agent is not allowed to not allocate any energy budget to the sensory actions). Each agent i is then faced with a budget–limited MAB such that the arms of the MAB are associated with the elements of $\mathfrak{A}_i$ (i.e. $\mathfrak{A}_i$ is the set of arms within the MAB model). For a particular arm $a := \left\langle n_i^{\mathrm{S}} e_i^{\mathrm{S}}, n_i^{\mathrm{Rx}} e_i^{\mathrm{Rx}}, n_i^{\mathrm{Tx}} e_i^{\mathrm{Tx}} \right\rangle$, the pulling cost of that arm is defined as $c := n_i^{\mathrm{S}} e_i^{\mathrm{S}} + n_i^{\mathrm{Rx}} e_i^{\mathrm{Rx}} + n_i^{\mathrm{Tx}} e_i^{\mathrm{Tx}}$. In addition, let B_i (i.e. the initial energy budget of agent i) be the budget of the budget–limited MAB. That is, within the MAB model, by choosing a particular action $a := \left\langle n_i^{\mathrm{S}} e_i^{\mathrm{S}}, n_i^{\mathrm{Rx}} e_i^{\mathrm{Rx}}, n_i^{\mathrm{Tx}} e_i^{\mathrm{Tx}} \right\rangle$ at time step t, agent i allocates energy budgets $B_i^{\mathrm{S}}(t) = n_i^{\mathrm{S}} e_i^{\mathrm{S}}$, $B_i^{\mathrm{Rx}}(t) = n_i^{\mathrm{Rx}} e_i^{\mathrm{Rx}}$, and $B_i^{\mathrm{Tx}}(t) = n_i^{\mathrm{Tx}} e_i^{\mathrm{Tx}}$ to data sampling, receiving, and transmission, respectively.

In contrast with the action set above, the definition of a single agent's reward function is not obvious. In particular, the reward function has to satisfy the requirement that if each agent maximises its own total rewards, then the agents together also maximise the total information collected in the network. However, in so doing, each agent has to take

into account the behaviour of other agents within the network as well. Thus, the reward function has to capture the affect of other agents' behaviour on the performance of a single agent. Given this, we develop a reward function for each agent i as follows. Recall that $\mathbf{S}_i(t)$, $\mathbf{Rx}_i(t)$ and $\mathbf{Tx}_i(t)$ are the set of sampled, received and transmitted data packets of agent i at time slot t. Furthermore, $\mathbf{Q}_i(t)$ is the set of total transmittable data packets in the memory (see Section 7.2.2 for more details). Let $\mathbf{Re}_i(t)$ denote agent i's set of residual packets from slot $(t-1)$ that are not transmitted until slot t. That is,

$$\mathbf{Re}_i(t) = \mathbf{Q}_i(t) / \mathbf{Tx}_i(t). \tag{7.16}$$

Given this, before we determine the reward function, let us consider the following informative case, where $\lambda = 1$; that is, there is no information decay as time passes by. Given this, throughout the operational time T of the network, the total information that is delivered to the BS is equal to the difference in the total information sampled by the agents in the network until time slot $(T-1)$, and the total amount of information that remains in the memory of the agents in the network at time slot T. In particular, since we assume that there is no data loss in our model, data sampled until time slot $(T-1)$ is either successfully delivered to the BS or still remains as residual data in the network at time slot T. Note that data sampled in time slot T is not considered here, since we assume that it cannot be delivered immediately to the BS, and as defined in Equations 7.16 and 7.3, $\mathbf{Re}_i(T)$ does not contain data that are sampled in time slot T. Thus, for each $t \in [1, T]$, let $r(t)$ denote the following function:

$$r_i(t) = \sum_{p \in S_i(t-1)} v(p, t-1) - \sum_{p \in Re_i(t)} v(p, t) + \sum_{p \in Re_i(t-1)} v(p, t-1). \tag{7.17}$$

Note that the first term on the right hand side of this equation is the total amount of sampled information of agent i at time slot $(t-1)$. The second term is the total information value of the residual data on agent i at time slot t, whilst the third term is the total information value of the residual data on agent i at time slot $(t-1)$. The intuition behind Equation 7.17 can be explained as follows. From the definitions given in Equations 7.3 and 7.16, the sum of the first and the third terms form the total amount of information that agent i can transmit in time slot t. In more detail, as we mentioned in Section 7.2.1, data sampled in time slot $(t-1)$ can only be transmitted from time slot t, and not earlier. Thus, the first term represents the total information content of this sampled data. The third term represents the amount of information that is not transmitted until time slot $(t-1)$. Both the sampled data and residual data, however, is available at time slot t for transmission. On the other hand, the second term represents the information value of data that is not sent by the end of time slot t, and thus, by subtracting it from the set of transmittable data (i.e. sum of previously sampled data and residual data from $(t-1)$), we get the throughput of agent i within time slot t. Given this, by using $r_i(t)$ as the reward function within the case of $\lambda = 1$, each part of agent i's chosen action (i.e. the chosen energy budgets) will effect the value of $r_i(t)$. In particular, the size of $B_i^{\mathrm{S}}(t)$ affects the total amount of sampled information, while $B_i^{\mathrm{Rx}}(t)$ and $B_i^{\mathrm{Tx}}(t)$ affect the size of residual data.

Now, we show that by maximising the sum of $r_i(t)$ over all t and i does indeed lead to the maximisation of the total amount of collected information within the network, in the case of $\lambda = 1$. In so doing, recall that $\sum_{p \in Re_i(t-1)} v(p,0) = 0$ for each agent i, since there is no residual data at all at the beginning. Given this, it is easy to see that if we sum up $r_i(t)$ by t from 1 to T, what we get as a result is exactly the difference of the total information collected by the network and the total amount of information that remains in the memory of the agents in the network. More precisely, we have

$$\begin{aligned}
\sum_{t=1}^{T} r_i(t) &= \sum_{t=0}^{T-1} \sum_{p \in S_i(t)} v(p,t) - \sum_{t=1}^{T} \sum_{p \in Re_i(t)} v(p,t) + \sum_{t=0}^{T-1} \sum_{p \in Re_i(t)} v(p,t) \\
&= \sum_{t=0}^{T-1} \sum_{p \in S_i(t)} v(p,t) - \sum_{p \in Re_i(T)} v(p,T) + \sum_{p \in Re_i(0)} v(p,0) \\
&= \sum_{t=0}^{T-1} \sum_{p \in S_i(t)} v(p,t) - \sum_{p \in Re_i(T)} v(p,T).
\end{aligned}$$

Recall that this value is equal to the total information that is succesfully delivered to the BS throughout the operation time of the network. Thus, $r_i(t)$ could be a possible reward function for agent i, since by maximising the total reward on interval $[0,T]$, the agents together also maximise the total amount of collected information value that is delivered to the BS as well.

Note that the definition of $r_i(t)$ in Equation 7.17 guarantees that in order to maximise the total amount of collected information, agent i cannot either ignore sampling, receiving or transmitting. In particular, for example, suppose that agent i ignores transmitting, and only focuses on just sampling/or receiving. In this case, the set of residual data at the end of time slot t is equal to the accumulated set of sampled data and residual data at time slot $(t-1)$, and thus, the value of the reward is 0. Now, it is easy to see that if the transmitting capacity is greater than 0 (i.e. $n_i^{\mathrm{Tx}}(t) > 0$), the reward value is definitely higher than 0 as well. In a similar vein, we can easily see that agent i cannot get high reward values in the long term if it ignores the other sensory tasks as well.

Now, to generalise Equation 7.17 to the case of $\lambda \neq 1$, consider the following:

$$R_i(t) = \lambda^{d_i - 1} \left\{ \sum_{p \in S_i(t-1)} v(p,t-1) - \sum_{p \in Re_i(t)} v(p,t) + \lambda \sum_{p \in Re_i(t-1)} v(p,t-1) \right\}, \qquad (7.18)$$

where d_i is the distance of agent i from the BS (in hops), and λ is the information durability coefficient. This equation differs from Equation 7.17 in two places. First, it is weighted by the factor $\lambda^{d_i - 1}$. The intuition behind using this factor is that since agent i

is d_i hops away from the BS, the information value that agent i transmits is decreased by a factor λ^{d_i-1} when the BS receives that data. The second difference is that the third term of Equation 7.18 is weighted with λ. The reason here is that since the third term represents the set of packets that are not sent by the end of time slot $(t-1)$, the information value of those packets is decreased in the next time slot. Note that in the case of $\lambda = 1$, this equation is reduced to Equation 7.17. To show that this reward function is suitable for maximising the total collected information of the network in the long term, we state the following:

Theorem 7.3. *Using the reward function defined in Equation 7.18, the total reward value that the agents in the WSN achieve together over the interval* $[0, T]$ *is equal to the total information content value delivered to the* BS *over that time interval.*

That is, Theorem 7.3 states that by maximising each agent's total reward over interval $[0, T]$, where the reward function is defined as in Equation 7.18, we can achieve the maximal information collected and delivered to the BS. We prove the theorem as follows:

Proof of Theorem 7.3. For the sake of simplicity, let $\mathfrak{L}_j$ denote the set of agents that are j hops from the BS. That is,

$$d_i = j, \forall i \in \mathfrak{L}_j. \tag{7.19}$$

Now, consider Equation 7.1 in Section 7.2.2. Note that since no data can be sampled and forwarded, or received and forwarded at the same time slot (see Section 7.2.1), no data packets are transmitted or received at time slot 0 in the whole WSN. Thus, using the notation of Section 7.2, the main objective can be rewritten as follows.

$$\max \sum_{t=1}^{T} \left\{ \sum_{p \in Rx_{BS}(t)} v\left(p, t\right) \right\}. \tag{7.20}$$

Consider a particular member of Equation 7.20, which is $\sum_{p \in Rx_{BS}(1)} v\left(p, 1\right)$. This equation determines the total information value that arrives to the BS at time slot 1. According to our assumptions in Section 7.2.1, no data loss occurs during any transmission. Thus, the amount of received information at the BS is equal to the total amount of information that is transmitted from agents that are 1–hop from the BS at time slot 1. That is,

$$\sum_{p \in Rx_{BS}(1)} v\left(p, 1\right) \quad = \quad \sum_{j \in \mathfrak{L}_1} \sum_{p \in Tx_j(1)} v\left(p, 1\right). \tag{7.21}$$

Note that the set of transmitted data of $\mathfrak{L}_1$ at time slot 1 is equal to the set of sampled data at time slot 0, excluding the set of residual data at time slot 1 (since there is no received data and the residual set is still empty at time slot 0). Since newly sampled data does not suffer from information value discounting, the right side of Equation 7.21

can be rewritten as the following:

$$\sum_{j\in\mathfrak{L}_1}\sum_{p\in Tx_j(1)} v\,(p,1) \;=\; \sum_{i\in\mathfrak{L}_1}\sum_{p\in S_i(0)} v\,(p,0) - \sum_{i\in\mathfrak{L}_1}\sum_{p\in Re_i(1)} v\,(p,1). \tag{7.22}$$

Now, let us consider the second member of Equation 7.20, which is $\sum_{p\in Rx_{BS}(2)} v\,(p,2)$. Similarly, this can be rewritten as follows.

$$\sum_{p\in Rx_{BS}(2)} v\,(p,2) \;=\; \sum_{j\in\mathfrak{L}_1}\sum_{p\in Tx_j(2)} v\,(p,2). \tag{7.23}$$

However, this is equal to the union of the set of received data, the set of sampled data, and the set of residual data at time slot 1, excluding the set of residual data of layer 1 at time slot 2. Furthermore, any of these sets may not be empty. The packets in the sets of received and residual data suffer from value discounting, thus, Equation 7.23 is equal to the following:

$$\begin{aligned}\sum_{p\in Rx_{BS}(2)} v\,(p,2) \;&=\; \sum_{j\in\mathfrak{L}_1}\sum_{p\in Tx_j(2)} v\,(p,2) = \\ &=\; \sum_{i\in\mathfrak{L}_1}\sum_{p\in S_i(1)} v\,(p,1) + \lambda\sum_{i\in\mathfrak{L}_1}\sum_{p\in Re_i(1)} v\,(p,1) + \\ &+\; \lambda\sum_{i\in\mathfrak{L}_1}\sum_{p\in Rx_i(1)} v\,(p,1) - \sum_{i\in\mathfrak{L}_1}\sum_{p\in Re_i(2)} v\,(p,2),\end{aligned} \tag{7.24}$$

where λ is the durability coefficient of the network. Now let us consider $\sum_{i\in\mathfrak{L}_1}\sum_{p\in Rx_i(1)} v\,(p,1)$. Similar to Equation 7.21, this can be written as:

$$\lambda\sum_{i\in\mathfrak{L}_1}\sum_{p\in Rx_i(1)} v\,(p,1) \;=\; \lambda\sum_{i\in\mathfrak{L}_2}\sum_{p\in Tx_i(1)} v\,(p,1). \tag{7.25}$$

Using Equations 7.24 and 7.25, and replacing $\mathfrak{L}_1$ with $\mathfrak{L}_2$ in Equation 7.22, we obtain the following:

$$\begin{aligned}\sum_{p\in Rx_{BS}(2)} v\,(p,2) \;&=\; \sum_{i\in\mathfrak{L}_1}\sum_{p\in S_i(1)} v\,(p,1) - \\ &-\; \sum_{i\in\mathfrak{L}_1}\sum_{p\in Re_i(2)} v\,(p,2) + \lambda\sum_{i\in\mathfrak{L}_1}\sum_{p\in Re_i(1)} v\,(p,1) + \\ &+\; \lambda\sum_{i\in\mathfrak{L}_2}\sum_{p\in S_i(0)} v\,(p,0) - \lambda\sum_{i\in\mathfrak{L}_2}\sum_{p\in Re_i(1)} v\,(p,1).\end{aligned} \tag{7.26}$$

In general, if we take the t^{th} member of Equation 7.20, then it can be decomposed as follows. If $t \le L$, where L is the number of the layers in the network, then:

$$\begin{aligned}\sum_{p\in Rx_{BS}(t)} v\,(p,t) \;&=\; \sum_{j=1}^{t}\lambda^{j-1}\sum_{i\in\mathfrak{L}_j}\sum_{p\in S_i(t-j)} v\,(p,t-j) - \\ &-\; \sum_{j=1}^{t}\lambda^{j-1}\sum_{i\in\mathfrak{L}_{j+1}}\sum_{p\in Re_i(t-j+1)} v\,(p,t-j+1)+\end{aligned}$$

$$+ \sum_{j=1}^{t} \lambda^{j} \sum_{i \in \mathfrak{L}_j} \sum_{p \in Re_i(t-j)} v\,(p, t-j). \tag{7.27}$$

Let us note that here $\sum_{i \in \mathfrak{L}_j} \sum_{p \in Re_i(0)} v\,(p,0) = 0$ for any layer j. That is, we can say that the amount of information that arrives to the BS at time slot t can be decomposed into the sum of data on layer 1 at time slot $(t-1)$, on layer 2 at time slot $(t-2)$, and so on. If $t > L$, however, the equation for this case is slightly different, since the decomposition stops at the last layer of agents. Thus, we have:

$$\begin{aligned} \sum_{p \in Rx_{BS}(k)} v\,(p,k) &= \sum_{j=1}^{L} \lambda^{j-1} \sum_{i \in \mathfrak{L}_j} \sum_{p \in S_i(t-j)} v\,(p, t-j) - \\ &- \sum_{j=1}^{L} \lambda^{j-1} \sum_{i \in \mathfrak{L}_{j+1}} \sum_{p \in Re_i(t-j+1)} v\,(p, t-j+1) + \\ &+ \sum_{j=1}^{L} \lambda^{j} \sum_{i \in \mathfrak{L}_j} \sum_{p \in Re_i(t-j)} v\,(p, t-j). \end{aligned} \tag{7.28}$$

Given this, combining Equations 7.27 and 7.28, and taking each t into account, we can reformulate our main objective to the following:

$$\begin{aligned} &\sum_{t=1}^{T} \left\{ \sum_{p \in Rx_{BS}(t)} v\,(p,t) \right\} \\ &= \sum_{t=1}^{T} \sum_{j=1}^{\min(t,L)} \lambda^{j-1} \sum_{i \in \mathfrak{L}_j} \left\{ \sum_{p \in S_i(t-j)} v\,(p, t-j) - \sum_{p \in Re_i(k-j+1)} v\,(p, t-j+1) + \lambda \sum_{p \in Re_i(k-j)} v\,(p, t-j) \right\}. \end{aligned} \tag{7.29}$$

Consider the core part of Equation 7.29 in the braces. Now, using the definition of the reward function in Equation 7.18 to replace that part, and recall that the distance of agent i is defined in Equation 7.19, we can reformulate 7.29 as follows:

$$\max \sum_{j=1}^{\min(T,L)} \sum_{t=0}^{T-j} \sum_{i \in \mathfrak{L}_j} R_i\,(t). \tag{7.30}$$

That is, the original objective can be decomposed to the sum of reward functions of agents on each layer j, from time slot 0 to time slot $T-j$. □

Now, using the aforementioned definitions of set of arms, pulling costs, budgets, and reward functions, the energy management problem of each agent i can be reduced to a budget–limited MAB problem. Thus, the multi–armed bandit based energy management algorithm works as follows. Each agent i uses a budget–limited MAB pulling algorithm in order to maximise its total reward over time. This can be done by using one of the pulling algorithms described in the previous three chapters of this thesis. Let us hereafter refer to this approach (i.e. using budget–limited MAB techniques for allocating energy budgets) as the *multi-armed bandit based energy management* (MAB/EM).

Note that within MAB/EM, the agents do not explicitly coordinate with each other (i.e. they do not use coordination messages). In more detail, our approach uses explicit communication messages within the routing part (for more details, see Section 7.4), but not within the energy budget allocation phase. However, these communication messages are only for evaluating the reward value of the chosen action (i.e. the chosen combination of energy budget allocations). Given this, the agents do not need to coordinate when they take an action. Despite the lack of explicit coordination within MAB/EM, the agents can still achieve coordination by only observing the reward value they get. In more detail, consider the definition of the reward function (Equation 7.18). Note that this reward function is affected by the agent's current chosen action (i.e. the energy amounts allocated to sampling, receiving and transmission). In particular, according to Equations 7.3 and 7.16, $\mathbf{Re}_i(t)$ (i.e. the list of residual packets) depends on the lists of sent and received packets, respectively. Thus, in order to achieve higher rewards, each agent aims to find actions that result in better reward values. However, the effectiveness of a chosen action also depends on other agents' action as well. Indeed, the effectiveness of data receiving (or transmitting) depends on the allocated budget to transmitting (or receiving) of other neighbouring agents. For example, it is not efficient for agent i to allocate a large amount of energy to receiving if its neighbours are only willing to send a small amount of data. Similarly, it is not efficient either for agent i to set large amount of energy to transmission when its neighbours can only receive a low number of packets. Note that in the latter case, by using MITRA (see Section 7.4 for more details), data loss will not occur. However, the amount of energy that agent i allocates to its data transmission will be lost (see Equation 7.8 in Section 7.2.3 for more detail). As a result, by only observing which actions result in higher rewards, the agents also learn to cooperate with the others as well. This implies that MAB/EM does not require large communication cost, and thus it satisfies Requirement 6 (i.e. limited use of communication). It also efficiently fulfils Requirements 5 (robustness and flexibility), since the agents do not depend on the size of the network. In addition, since this approach uses the budget–limited approach to learn the optimal energy allocation settings that maximise the long–term information collection, it fulfil Requirement 4 (adaptivity) as well.

7.3.2 Computational Complexity Analysis

Since WSNs are heavily resource constrained (i.e. the low energy capacity, small size and tight computational constraints), algorithms that are implemented for such networks need to take into consideration the limited computational capacity and memory space (Akyildiz *et al.*, 2002; Rogers *et al.*, 2009). Thus, in order to ensure that MAB/EM is suitable for WSNs (i.e. it can be installed to real sensors), we have to guarantee that it has low computational complexity and low memory demand. Given this, we study

the performance of the MAB/EM in terms of computational complexity in this section. More precisely, we investigate the number of computational steps (i.e. running time cost) and the memory usage that MAB/EM uses at each time slot.

From the aspect of computational cost, by using the budget–limited MAB algorithms from the previous three chapters, each agent i can have the following total computational complexity: (i) $O\left(\varepsilon \left|\mathfrak{A}_i\right| B_i + \left|\mathfrak{A}_i\right| \ln \left|\mathfrak{A}_i\right|\right)$ (by using the budget–limited ε–first approach); (ii) $O\left(B_i \left|\mathfrak{A}_i\right|\right)$ (by using fractional KUBE or fractional KDE); and (iii) $O\left(B_i \left|\mathfrak{A}_i\right| \ln \left|\mathfrak{A}_i\right|\right)$ (by using KUBE or KDE). Note that $\left|\mathfrak{A}_i\right|$ is the number of arms within the budget–limited MAB model of agent i. Since the size of the operating time interval is proportional to the budget size B_i, it is easy to show that the average computational cost (i.e. cost per time step) of agent i is: (i) $O\left(\varepsilon \left|\mathfrak{A}_i\right|\right)$; (ii) $O\left(\left|\mathfrak{A}_i\right|\right)$; and (iii) $O\left(\left|\mathfrak{A}_i\right| \ln \left|\mathfrak{A}_i\right|\right)$. Note that $\left|\mathfrak{A}_i\right|$ typically has the value of at most few thousands. This can be easily calculated by using the typical sensory parameter values, which can be found, for example, in Kansal and Srivastava (2003). This implies that the computational cost is low in all the aforementioned cases.

In terms of memory usage, MAB/EM is also efficient. In particular, recall that each agent i either uses the density–ordered greedy algorithm or the fractional relaxation method to estimate the best combination of arms at each time step. We can efficiently run both methods with at most $O\left(\left|\mathfrak{A}_i\right|\right)$ memory place (Cormen *et al.*, 2001). They also need $O\left(\left|\mathfrak{A}_i\right|\right)$ to maintain the parameters of each arm (i.e. mean value, number of pulls, or its ranking). Given this, the memory usage of MAB/EM is $O\left(\left|\mathfrak{A}_i\right|\right)$. To demonstrate that the memory usage is indeed low, compared to the size of data packets, consider the following example. Suppose that to store a number, each agent uses 4 bytes of memory. Using the fact that $\left|\mathfrak{A}_i\right|$ is typically at most few thousands, the total memory usage (i.e. to store the arrays of probability and weight parameters) is at most few kilobytes. This is small, compared to the total size of real data that the agents typically have to forward in many applications (e.g. in wireless visual sensor networks) the average size of a single data packet is likely to be $10 - 100$ kBytes (Kho *et al.*, 2010).

7.4 Optimal Data Routing

Given the energy management approach described in the previous section, we now focus on the maximal information throughput problem presented in Section 7.2.4. Thus, this section outlines the work undertaken towards addressing this routing problem. Specifically, here we describe two decentralised algorithms that allow agents to achieve maximal information throughput between neighbouring layers, with respect to their energy constraints. In particular, the first algorithm, called MITRA (for maximal information throughput routing algorithm), achieves optimal performance in terms of solving

the maximal information throughput problem. However, it can have significant computational and communication costs in some settings. On the other hand, the second algorithm, called MITRA$_\tau$, produces near–optimal performance (approximately 98% of the optimal performance), but with reduced communication and computational costs. To this end, we first introduce MITRA in more detail in Section 7.4.1. Following this, we show that this approach is optimal in terms of maximising the information throughput in Section 7.4.2. Furthermore, we provide a theoretical upper bound for the computational and communication costs of MITRA in Section 7.4.3. Finally, we propose MITRA$_\tau$, a modified version of MITRA with reduced communication and computational costs in Section 7.4.4.

7.4.1 The Maximal Information Throughput Routing Algorithm

Recall that at each time slot t, all the agents within the system run the MAB/EM in order to set up the energy budgets for that current time slot. Then, their next step is to maximise the amount of forwarded information value conditional on the budgets in that given time slot. That is, the agents aim to maximise the total information value forwarded between neighbouring layers of agents (see definition 7.2 for more details). Now, let $\mathfrak{L}_l$ and $\mathfrak{L}_{l-1}$ denote the corresponding layers. The pseudocode of the MITRA run by the agents within these layers is depicted in Algorithm 7.1.

In more detail, we refer to the agents in layers $\mathfrak{L}_l$ and $\mathfrak{L}_{l-1}$ as senders, and receivers, respectively. The algorithm can be outlined as follows:

- **Step 3**: First, each sender s_i broadcasts a message that contains the list of 2–tuples to each of its neighbouring receivers. The first element of the tuple contains the packet ID, whilst the second element contains the information value of sender s_i's transmittable packets (i.e. the list of $\mathbf{Q}_{s_i}(t)$, see Section 7.2.2 for more details). Then, whilst data transmission is still feasible, the algorithm repeatedly executes steps $5-10$ as follows.

- **Step 5**: Based on the received information lists from the neighbouring senders, each receiver r_j chooses the best packets (i.e. packets with the highest information value) it can receive, with respect to its residual receiving capacity (i.e. the maximal number of packets it can still receive without exceeding its total receiving capacity $N_{r_j}^{\mathrm{Rx}}$). Note that $N_{r_j}^{\mathrm{Rx}}$ is set by the MAB/EM (see Section 7.3.1 for more details). In so doing, it needs to wait until it receives all the broadcast information from its neighbouring senders. However, since node failures may occur, agent r_j does not exactly know which of its neighbours is available within the current time slot t, and thus, will send to r_j a broadcast message. In such cases, r_j does not know when to stop waiting for the broadcast messages, and thus, it cannot move

Algorithm 7.1 MITRA

1: **for all** pair of layers $\mathfrak{L}_l$ and $\mathfrak{L}_{l-1}$ **do**
2: agents in layer $\mathfrak{L}_l \leftarrow$ senders, agents in layer $\mathfrak{L}_{l-1} \leftarrow$ receivers;
3: $\forall i$ sender s_i broadcasts list of information values;
4: **while** data transmission is feasible **do**
5: $\forall j$: when receiver r_j receives all the broadcast information (or time threshold expires), it identifies best packets it can receive;
6: $\forall j$ receiver r_j sends `REQUEST` messages to senders;
7: $\forall i$ when sender s_i receives all the `REQUEST` messages (or time threshold expires), it sends data to receiver with best offer;
8: **if** $\exists$ sender s_i has not exceed transmission budget **then**
9: sender s_i broadcasts a `SEND` message to receivers;
10: **end if**
11: **end while**
12: **end for**

on to the next step of MITRA. In order to avoid this situation, we set a time threshold, so that if this threshold expires, the sender stops waiting for further broadcast messages. Following this, r_j chooses the best packets it can receive as follows. It first sorts the received lists of 2–tuples in decreasing order of the value of information, then it merges these lists into a joint list, also with the decreasing order of the information value. From this joint list, it chooses the best packets it can receive.

- **Step 6**: Following this, receiver r_j propagates `REQUEST` messages to each of its neighbouring senders. In particular, each `REQUEST` message contains the number of packets that r_j requests from that sender. This number is calculated in step 5 of the algorithm.

- **Step 7**: When s_i receives all the `REQUEST` messages from its neighbouring receivers, it chooses the best offer; that is, the one with the highest number of requested packets. However, similarly to step 5 of the algorithm, it may occur that s_i does not know when to stop waiting for all the `REQUEST` messages, due to node failure. Thus, to prevent it from waiting indefinitely for the messages, we also use a time threshold here. Given this, after all the `REQUEST` messages arrive to s_i, or the time threshold expires, s_i sends the requested packets to the receiver with the best offer. If the receiver with the best offer is not unique, then s_i randomly chooses one among them.

- **Steps 8–10**: After data transmission in the previous step, if sender s_i still has the capacity to transmit data (i.e. $n_{s_i}^{\mathrm{Tx}}(t)$ is not exceeded), then it broadcasts a `SEND` message to each of its neighbouring receivers. This message contains the number of packets that it transmitted in step 7. Based on this message, all the receivers can update the list of packets they can request from s_i (i.e. they update the joint list described in step 3). Furthermore, they also update the value of their remaining receiving capacity.

Now, to detect whether data transmission is still feasible, the participating agents do the following. From the sender side, when sender s_i does not receive any `REQUEST` messages in step 7, it considers data transmission as not feasible. From the receiver side, when receiver r_j does not receive any broadcast messages (e.g. the list of information value, or the `SEND` messages) in step 5, then it also considers data transmission as not feasible. Given this, if an agent sees that it cannot receive and transmit data anymore (i.e. receiving and transmission is not feasible), it stops running MITRA for that time slot. That is, the agents rerun MITRA at each time slot t. Note that the time thresholds in steps 5 and 7 are for only communication messages (i.e. `REQUEST` and broadcast messages). Once the agent receives one of these messages from its corresponding neighbour,

it sets up a communication channel, in which data packets are assumed to be successfully forwarded, without any loss.

7.4.2 Performance Analysis

Given the description of MITRA above, we now show this algorithm provides the optimal solution to the maximal information throughput routing problem presented in Definition 7.2. In so doing, we state the following:

Theorem 7.4. *Assuming that the communication between senders and receivers is perfect, that is, none of the messages arrive after the timeout, the MITRA algorithm results in an optimal solution for the maximal information throughput routing problem (i.e. the solution that gives the maximal throughput of information value between the sender and receiver layers).*

Proof. Here we use the contradiction technique. Let us assume that the MITRA algorithm given in the previous section is not optimal. That is, the output solution does not maximise the total transmitted information value between the two layers. Let $\mathfrak{O}$ denote the output solution of the MITRA algorithm and $\mathfrak{O}_{OPT}$ be one of the optimal solutions. Since we assume that $\mathfrak{O}$ is not optimal, there should be p_1 and p_2 packets such that only one of them is allocated in $\mathfrak{O}$ and the other one is allocated in $\mathfrak{O}_{OPT}$. Without loss of generality, we can assume that p_1 is allocated in $\mathfrak{O}$ and p_2 is allocated in $\mathfrak{O}_{OPT}$. We can also assume that both p_1 and p_2 are sent to the same receiver r_j. It is easy to prove that if $\mathfrak{O} \neq \mathfrak{O}_{OPT}$ then there exist two packets such that these assumptions hold.

In particular, there are two cases to investigate. In the first, both p_1 and p_2 are from the same sender. Note that it is easy to show that $v(p_1, k) \geq v(p_2, k)$. That is, p_1 has a higher information value than p_2, since the corollary states that those data which are sent from the sender must be the packets with the highest values in the set of packets of that sender.

In the second case, p_1 and p_2 are from different senders. Since in MITRA, the receiver uses a greedy approach to allocate possible arriving packets, when p_1 is accepted and p_2 is not at r_J, the only explanation is that $v(p_1, k) \geq v(p_2, k)$.

One can see that in both cases p_1 has a higher, or at least the same value, as p_2. If p_1 has a higher value than that of p_2, then by replacing p_2 in $\mathfrak{O}_{OPT}$ with p_1, we would have a better solution than $\mathfrak{O}_{OPT}$. However, this is a contradiction, since $\mathfrak{O}_{OPT}$ is assumed to be optimal. If p_1 has the same value as p_2, then by replacing all the possible p_i-s that are in $\mathfrak{O}$ but not in $\mathfrak{O}_{OPT}$ (since they all have the same value, otherwise we would be faced with the former case), we would have that $\mathfrak{O}$ is also an optimal solution, which

would also contradict our assumption at the beginning. Therefore one can see that the original assumption, that is, $\mathfrak{O}$ is not optimal, is not true. □

7.4.3 Computational and Communication Cost of MITRA

In the previous section, we showed that MITRA achieves an optimal solution for the maximal information throughput problem. Given this, here we continue the analysis of MITRA by studying its computational and communication cost. In particular, similarly to the case of MAB/EM, we need to analyse whether MITRA is efficient in terms of computational and communication complexity. In so doing, recall that at each time slot t, each agent i within the network repeatedly runs steps $4-11$ of Algorithm 7.1 until data transmission is not feasible at that time slot. For the sake of simplicity, hereafter we refer to this cycle as the *communication round* of MITRA (since the agents communicate with each other during this cycle in order to find the maximal information throughput). Note that since MITRA is rerun at every time slot, each time slot t contains a number of communication rounds. Thus, the number of communication rounds that MITRA uses within a particular time slot cannot be larger in time, compared to the length of a single time slot. Given this, here we aim to analyse whether we can upper bound the number of communication rounds. Furthermore, note that both the computational and communication costs of agent i depend on the number of communication rounds that the agent needs to run. Thus, in order to guarantee low computational and communication costs of a single agent, we also need to ensure that the number of communication rounds that an agent uses within the MITRA is also low. In more detail, each receiver determines the best packets (i.e. packets with highest information value) it can receive by sorting the list of receivable packets at each communication round (step 5 of Algorithm 7.1). Since this list typically has a size at most of few thousands, sorting it is simple and fast (e.g. by quicksort). However, since the sorting is repeatedly executed at each communication round, if the number of those rounds is high, then the total computational cost can be significant. Now, note that the communication cost of a single agent consists of the cost of sending `REQUEST` messages and the cost of sending a `SEND` broadcast message at each communication round. Thus, again, if the number of communication rounds is high, then the total communication cost can also be significant.

Against this background, we provide a worst–case upper bound (i.e. an upper bound that holds for all the cases) for the number of communication rounds that MITRA uses. More precisely, we state the following:

Theorem 7.5. *Consider neighbouring layers $\mathfrak{L}_l$ and $\mathfrak{L}_{l-1}$. At each time slot t, let $T_{\mathrm{com}}(t)$ denote the total number of communication rounds, that MITRA needs to run until data transmission is not feasible between layers $\mathfrak{L}_l$ and $\mathfrak{L}_{l-1}$ within time slot t.*

Given this, we have:

$$T_{\text{com}}(t) \leq \frac{\ln\left(\sum_{r_j \in \mathfrak{L}_{l-1}} N_{r_j}^{\text{Rx}}\right)}{\ln|\mathfrak{L}_{l-1}| - \ln\left(|\mathfrak{L}_{l-1}| - 1\right)},$$

where $|\mathfrak{L}_{l-1}|$ *denote the size of layer* $\mathfrak{L}_{l-1}$ *(i.e. layer of receivers).*

Proof. Recall that, at each communication round, each receiver r_j chooses the best packets it can receive, conditional to the value of its residual receiving capacity (see step 5 of algorithm 7.1). Let $D_{r_j}(\tau)$ denote the maximal number of packets r_j can receive from its neighbouring senders at communication round τ. It is easy to see that for each r_j, $D_{r_j}(\tau)$ is monotone decreasing function of τ, within time slot t. In more detail, recall that the senders cannot forward information that are sampled or received at time slot t. Given this, $D_{r_j}(\tau)$ only contains data that are sampled/or received until time slot $(t-1)$. This set of data, however, is already given at the beginning of time slot t, and thus, during the communication rounds, the size of these data cannot be increased. Furthermore, at each communication round (within time slot t), receiver r_j receives a non–negative number of packets. Given this, the value of $D_{r_j}(\tau)$ is monotone decreasing.

Given this, we first show that at each communication round τ, the total number of successfully received packets within MITRA is at least $D_{\max}(\tau)$, where

$$D_{\max}(\tau) = \max_{r_j} D_{r_j}(\tau).$$

Indeed, according to algorithm 7.1, each receiver r_j send `REQUEST` messages to its neighbours at each communication round τ, requesting $D_{r_j}(\tau)$ packets in total. Some of these requests will be accepted by the senders, whilst the others will be rejected. However, a sender only rejects a request, if it gets a better request (or a same request) of total amount of information value from another receiver. This implies that the number of packets of the better request is not lower than the number of packets r_j requests from that sender. Given this, it is easy to see that the total amount of transmitted (received) packets is at least $D_{r_j}(\tau)$ for any r_j (i.e. it is also at least $D_{\max}(\tau)$). Therefore, we have the following inequality:

$$\sum_{r_j} D_{r_j}(\tau+1) \leq \sum_{r_j} D_{r_j}(\tau) - D_{\max}(\tau). \tag{7.31}$$

Now, note that at each communication round τ, we have:

$$D_{\max}(\tau) \geq \frac{\sum_{r_j} D_{r_j}(\tau)}{|\mathfrak{L}_{l-1}|}. \tag{7.32}$$

That is, $D_{\max}(\tau)$ is not lower than the average value of $D_{r_j}(\tau)$. Using Equations 7.31 and 7.32, we get:

$$\sum_{r_j} D_{r_j}(\tau+1) \leq \frac{|\mathfrak{L}_{l-1}|-1}{|\mathfrak{L}_{l-1}|} \sum_{r_j} D_{r_j}(\tau).$$

That is, we can show by induction that the following holds for each τ:

$$\sum_{r_j} D_{r_j}(\tau+1) \leq \left(\frac{|\mathfrak{L}_{l-1}|-1}{|\mathfrak{L}_{l-1}|}\right)^{\tau} \sum_{r_j} D_{r_j}(1). \tag{7.33}$$

Note that $D_{r_j}(1) \leq N_{r_j}^{\mathrm{Rx}}$; that is, the maximal number of packets that r_j can receive at the first communication round is not greater than the receiving capacity of r_j. Given this, from Equation 7.33 we get:

$$\sum_{r_j} D_{r_j}(\tau+1) \leq \left(\frac{|\mathfrak{L}_{l-1}|-1}{|\mathfrak{L}_{l-1}|}\right)^{\tau} \sum_{r_j} N_{r_j}^{\mathrm{Rx}}. \tag{7.34}$$

Now, note that MITRA stops after τ communication rounds if and only if

$$\sum_{r_j} D_{r_j}(\tau+1) < 1.$$

That is, no more packets can be sent to the receivers. Given this, MITRA still runs after τ communication rounds if

$$\left(\frac{|\mathfrak{L}_{l-1}|-1}{|\mathfrak{L}_{l-1}|}\right)^{\tau} \sum_{r_j} N_{r_j}^{\mathrm{Rx}} \geq 1. \tag{7.35}$$

This can be reformulated as:

$$\sum_{r_j} N_{r_j}^{\mathrm{Rx}} \geq \left(\frac{|\mathfrak{L}_{l-1}|}{|\mathfrak{L}_{l-1}|-1}\right)^{\tau}. \tag{7.36}$$

Taking the logarithmic function of both sides, we get:

$$\ln\left(\sum_{r_j} N_{r_j}^{\mathrm{Rx}}\right) \geq \tau\left(\ln|\mathfrak{L}_{l-1}| - \ln\left(|\mathfrak{L}_{l-1}|-1\right)\right). \tag{7.37}$$

Substituting $T_{\mathrm{com}}(t)$ into this inequality concludes the proof. □

Note that from the proof, it is easy to show that this upper bound is tight. Thus, $T_{\mathrm{com}}(t) = O\left(\ln\left(\sum_{r_j \in \mathfrak{L}_{l-1}} N_{r_j}^{\mathrm{Rx}}\right)\right)$; that is, the upper bound of T_{com} is the logarithm of the total number of packets that need to be forwarded within each time slot t.

7.4.4 Communication Round Limited MITRA

In the previous section, we provided an upper bound for the number of communication rounds that MITRA uses. In particular, we demonstrated that the number of these communication rounds is low, compared to the total size of data to be forwarded at a single time slot. However, since this upper bound is tight, the total number of communication rounds that MITRA uses in the worst case scenario (i.e. when the bound is tight) is still significant in terms of total time length. For example, consider a WSN, where each layer has 10 agents on average, and each agent can receive 100 packets per time slot. Given this, according to Theorem 7.5, the upper bound of the number of communication rounds is around 66. Note that each communication round consumes a certain amount of time, and thus, 66 communication rounds together may not fit into the length of a single time slot (since MITRA has to terminate within the same time slot).

In order to address this shortcoming, we can either shorten the time length of a communication round, risking the higher rate of data loss in WSNs (i.e. not all of the `SEND` and `REQUEST` messages arrive on time), or limit the number of communication rounds that MITRA can use. We show that by using the latter solution, we can significantly reduce the number of communication rounds, whilst the reduction in the performance of the algorithm is not significant. We denote the communication round limited MITRA with MITRA$_\tau$, where τ is the threshold value of the number of communication rounds. Given this, the algorithm for MITRA$_\tau$ is similar to that of MITRA, except that it stops executing steps $4-11$ after exactly τ rounds (see Algorithm 7.1 for more details). In Section 7.5.4, we will demonstrate that with low τ values (e.g. $\tau = 8$), MITRA$_\tau$ can still achieve 98% of MITRA's performance.

7.5 Performance Evaluation

Having calculated the computational and communication complexity of MAB/EM and MITRA in the previous sections, we now demonstrate that by using MAB/EM for energy management and MITRA$_\tau$ for data routing, our proposed algorithms together significantly outperform the state–of–the–art. In particular, we first compare the performance of different MAB/EM techniques, that uses the budget–limited MAB algorithms from the previous chapters to tackle the energy management problem. With this comparison, we study which of the algorithms efficiently fulfil Requirement 1 (i.e. good experimental performance quality). The reason we choose MITRA$_\tau$ instead of MITRA to route data is that the communication cost of MITRA$_\tau$ is guaranteed to be low (see Section 7.4.4 for more details). However, as we will show later, it achieves, on average, 98% of the performance of MITRA.

Following this, we present empirical results against state–of–the–art algorithms that demonstrates the efficiency of using budget–limited MAB in long–term information collection within the WSN domain. In so doing, we need to choose a benchmark algorithm that has to fulfil the following requirements:

- It must be capable of using efficient adaptive sampling methods for collecting data from the environment.
- It must use information content valuation, in order to distinguish important data from unimportant data.
- It must contain an energy management policy, which allocates energy budgets to different sensory tasks of sampling, receiving, and transmitting.

In particular, as we discussed in Section 7.1, algorithms that guarantee these requirements may perform well in WSNs with dynamic environments for efficient long–term information collection. On the other hand, those which fail to fulfil the aforementioned requirements are not suitable for long–term information collection in our settings (see Section 7.1 for more details). As we demonstrated within Section 7.1, USAC is the most appropriate state–of–the–art method that fulfils these criteria. Given this, we choose USAC as a benchmark for our performance evaluation. In more detail, we compare the performance of our approach to USAC through extensive simulations, and we show that our approach typically outperforms USAC by around 120% on average in terms of long–term information collection. Furthermore, we also benchmark the performance of our approach against a non–learning approach, that solely uses MITRA for routing. In particular, within this benchmark approach, each agent randomly chooses an energy budget allocation combination, that it uses throughout its operating time (i.e. the budgets are fixed over time). Here, MITRA with fixed budgets represents a benchmark algorithm that does not intelligently set the budgets of the sensory tasks to adapt to the environmental changes. With this comparison, we demonstrate that by using adaptive learning (i.e. the MAB/EM), we can also achieve 100% improvement of collected information in the long term.

In addition, we also benchmark the performance of our approach against a centralised algorithm, that has the perfect knowledge of the environment, such as: the real value of any possible data in the future, the current energy level of each agent node, and whether they are out of order (i.e. suffering from node failure). Since this approach has perfect information of the future, we can This benchmark aims to provide a theoretical upper bound of the performance that we can achieve within long–term information collection in WSNs. In particular, in order to determine the optimal performance of the network, we need global information about each agent's sampled information values at each time slot. However, to gather this global information, a centralised control mechanism is

needed, which is not feasible in our settings (as outlined in Section 1.2). Thus this is a benchmark algorithm only; not a feasible solution to our information collection problem.

Finally, we demonstrate that by using MITRA$_\tau$ with small values of τ, we can still achieve near–optimal routing performance, while the number of communication rounds needed is significantly reduced (compared to that of the MITRA).

To this end, in Section 7.5.1, we first set the parameters that will be used throughout our simulations. We continue with the performance comparison between different budget–limited MAB approaches for the energy management problem (Section 7.5.2). Following this, to demonstrate the efficiency of MAB/EM combined with MITRA$_\tau$, we analyse simulation results in detail in Section 7.5.3. Here, we compare the performance of our approach to that of USAC, and the centralised optimal algorithm. Finally, in Section 7.5.4, we show that by using a small value of τ, MITRA$_\tau$ achieves near optimal performance (e.g. 98% of the optimal solution can be achieved with $\tau = 8$).

7.5.1 Parameter Settings

To compare the performance of the algorithms, we measure the overall amount of information collected by each algorithm over time. To this end, we run each algorithm on several networks with different topologies and environmental characteristics (e.g. the occurrence frequency of the events, or the expected value of information of each event). Then, we take the average of the specific results of the networks. In order to do this, we have to create a number of networks that may differ from each other in both topology and environmental characteristics. Given this, we now describe the parameter settings, that are used throughout our simulations, in order to create these networks and their environments.

In our model, a data packet that agent i samples from the environment has the information value randomly chosen from a normal distribution with mean m_i, variance v_i. To ensure positive information value, the distribution is truncated at 0 and $2m_i$. The value of each (m_i, v_i) pair is randomly and independently chosen from intervals $1-5$, and $1-3$, respectively. These values are set to be fixed over each simulation round. In addition, we tune these values such that nodes that are closer to the BS has lower mean values. This assumption is reasonable, since it reflects the fact that events farther from the BS are typically more important to the system.

Now, we set the energy settings of each agent node as follows. Each sensor's transmission, receiving and sampling energy consumption is uniformly and randomly chosen from intervals of $30-42$, $20-34$, and $15-25$ per packet, respectively. At each time step, the threshold values for the maximal number of packets, that agent i can sample, receive, and transmit, are set to be between 5 and 15. In addition, the battery capacity of each

agent node varies between $1.5 \cdot 10^6$ and $1.8 \cdot 10^6$. Note that these values are proportional to real–world sensor values as reported in Kansal and Srivastava (2003) and Torah *et al.* (2008). Given this, in our simulations, we use these values to set the parameters, such as e_i^{S}, e_i^{Tx}, e_i^{Rx}, and B_i, of the agents. We assume that the network contains 100 agents, and we randomly set the number of nodes in the layers such that each layer cannot contain more than 10 nodes. The communication edges of the network are randomly generated with probability 0.5 (i.e. two nodes within neighbouring layers can communicate with each other with probability 0.5). Note that we randomly set the corresponding values of the agents at the beginning of each simulation round, we set them to be fixed over that simulation round.

Now, note that within this chapter, we focus on long–term information collection, and thus, we do not have strict constraints on the delivery time of each collected piece of information (see Section 1.2 for more details). Given this, the information durability factor that we consider here is typically close to 1 (see Section 7.2.1). However, it would be also interesting to study the performance of our approach in systems where real–time information collection is desired. Within these systems, the real–time monitoring typically requires newest data only, and thus, the value of sampled information rapidly decreases as time passes by. This indicates that the durability factor is significantly lower within such systems. Now, note that MITRA does not have any guarantee that it will deliver the sampled data to the BS within a certain time threshold (which is a key requirement in real–time monitoring systems). Given this, our hypothesis is that our approach may not perform well in systems that demand low durability factors. In order to evaluate this hypothesis in more detail, we vary the value of λ during our simulations. In particular, we set the information durability coefficient $\lambda = 0.9$, and 0.5, respectively. The former represents the durability factor of non real–time systems, while the latter is a typical value for real–time WSNs.

In addition, we allow node failures during the operation of the WSN. In particular, each agent node may stop functioning at each time step with probability 0.2, independently from other nodes. Nodes with failures may be functioning again in the next time step. By allowing node failure, our hypothesis is that the performance of EM/MAB is significantly decreased, since the proposed budget–limited MAB algorithms cannot deal with non-stationary (i.e. dynamic) environments. As a result, we set up three simulation scenarios as follows:

1. Static topology (i.e. there is no node failure), and the information durability coefficient $\lambda = 0.9$.
2. Dynamic topology (i.e. node failure is allowed within the network), and $\lambda = 0.9$.
3. Dynamic topology (i.e. node failure is allowed within the network), but $\lambda = 0.5$.

Algorithms	Static topology ($\lambda = 0.9$)	Dynamic topology ($\lambda = 0.9$)	Dynamic topology ($\lambda = 0.5$)
Budget–limited ε–first ($\varepsilon = 0.05$)	$\mathbf{14.2}(\pm\mathbf{0.77}) \cdot \mathbf{10^5}$	$\mathbf{6.56}(\pm\mathbf{0.43}) \cdot \mathbf{10^5}$	$2.31(\pm0.26) \cdot 10^5$
Budget–limited ε–first ($\varepsilon = 0.10$)	$12.2(\pm0.85) \cdot 10^5$	$6.03(\pm0.61) \cdot 10^5$	$\mathbf{2.34}(\pm\mathbf{0.39}) \cdot \mathbf{10^5}$
Budget–limited ε–first ($\varepsilon = 0.15$)	$9.84(\pm0.91) \cdot 10^5$	$5.87(\pm0.47) \cdot 10^5$	$1.93(\pm0.47) \cdot 10^5$
KUBE	$\mathbf{8.35}(\pm\mathbf{0.91}) \cdot \mathbf{10^5}$	$\mathbf{4.26}(\pm\mathbf{0.36}) \cdot \mathbf{10^5}$	$\mathbf{1.67}(\pm\mathbf{0.22}) \cdot \mathbf{10^5}$
Fractional KUBE	$\mathbf{5.35}(\pm\mathbf{0.77}) \cdot \mathbf{10^5}$	$\mathbf{2.84}(\pm\mathbf{0.3}) \cdot \mathbf{10^5}$	$\mathbf{1.46}(\pm\mathbf{0.18}) \cdot \mathbf{10^5}$
KDE ($\gamma = 50$)	$8.99(\pm0.99) \cdot 10^5$	$5.18(\pm0.58) \cdot 10^5$	$1.78(\pm0.27) \cdot 10^5$
KDE ($\gamma = 100$)	$\mathbf{10.9}(\pm\mathbf{1.05}) \cdot \mathbf{10^5}$	$5.49(\pm0.42) \cdot 10^5$	$\mathbf{2.06}(\pm\mathbf{0.33}) \cdot \mathbf{10^5}$
KDE ($\gamma = 150$)	$9.3(\pm0.85) \cdot 10^5$	$\mathbf{5.62}(\pm\mathbf{0.44}) \cdot \mathbf{10^5}$	$1.99(\pm0.34) \cdot 10^5$
Fractional KDE ($\gamma = 50$)	$\mathbf{8.77}(\pm\mathbf{0.94}) \cdot \mathbf{10^5}$	$3.17(\pm0.45) \cdot 10^5$	$1.69(\pm0.31) \cdot 10^5$
Fractional KDE ($\gamma = 100$)	$7.48(\pm1.01) \cdot 10^5$	$3.53(\pm0.39) \cdot 10^5$	$\mathbf{1.83}(\pm\mathbf{0.25}) \cdot \mathbf{10^5}$
Fractional KDE ($\gamma = 150$)	$6.88(\pm0.75) \cdot 10^5$	$\mathbf{4.23}(\pm\mathbf{0.45}) \cdot \mathbf{10^5}$	$1.81(\pm0.29) \cdot 10^5$

TABLE 7.1: Total collected information with different budget–limited MAB algorithms.

In more detail, within the first scenario, MAB/EM has to deal with a static environment, while in the second scenario, it has to take the varying topology into account as well. In addition, within the third scenario, the system is forced to deliver the packets to the BS as fast as possible, since the information value of the packets rapidly converges to 0.

Finally, we run the simulations until the network cannot collect any further data (i.e. data that are collected and delivered to the BS).

7.5.2 Overall Performance Evaluation

Given the parameter settings above, we now discuss the numerical results of the simulations in more detail. In particular, we first study the performance of MAB/EM with different budget–limited MAB algorithms, combined with MITRA_8 (i.e. $\tau = 8$). As we will show later in Section 7.5.4, the choice of $\tau = 8$ results in both low performance loss and low number of communication rounds within MITRA.

Within the simulations, we set the budget–limited ε–first approach with different values of ε, namely $0.05, 0.1$, and 0.15, respectively. We also vary the value of γ to be $50, 100$, and 150 within KDE and its fractional counterpart. The results are depicted in Table 7.1

and Figure 7.1. In particular, Table 7.1 depicts the average total amount of collected information within the WSN by using different budget–limited MAB techniques. We highlight the best performance of algorithms that were run with different tuning parameter values (e.g. ε or γ). For the sake of better comparison, we also highlight the performance of KUBE and that of its fractional counterpart. It can be clearly seen that the budget–limited ε–first approach has a significantly better performance, compared to that of the others. In particular, it outperforms KUBE and fractional KUBE by up to 70% and 160%,while it is typically better than KDE and fractional KDE by up to 30% and 85%, respectively. In contrast, KUBE and its fractional counterpart provides the lowest performance in general. As a result, in terms of satisfying Requirement 1 (good experimental performance quality), the budget–limited ε–first approach has the highest performance, while KUBE and fractional KUBE performs the worst. That is, the decreasing ε–greedy based algorithms (i.e. KDE and fractional KDE) can be regarded as a trade–off approach that efficiently balances theoretical requirements with empirical criteria. In particular, as we showed in the previous chapters, KDE and fractional KDE achieves asymptotically optimal regret bounds. In addition, it provides adequately close empirical performance to that of the budget–limited ε–first approach.

Note that by using the density–ordered greedy approach to solve the underlying knapsack problem at each time step, we can improve the performance of the UCB based and the decreasing ε–greedy based algorithms, compared to the case when we use the fractional relaxation approach. In particular, this improvement is typically 70% in the case of KDE, and is approximately 50% in the case of KUBE, respectively.

To better understand the behaviour of each budget–limited MAB algorithm, we depict their performance over time in Figure 7.1. In more detail, we depict the performance of the KUBE, fractional KUBE, the budget–limited ε–first with $\varepsilon = 0.05$, and KDE and its fractional counterpart, both with $\gamma = 100$. The reason behind the choice of these values is that they typically outperform other choices of ε, and γ, respectively (see Table 7.1 for more details). The performance of the algorithms are measured in networks with: (i) static topology and $\lambda = 0.9$ (Figure 7.1a); (ii) dynamic topology and $\lambda = 0.9$ (Figure 7.1b); and (iii) dynamic topology and $\lambda = 0.5$ (Figure 7.1c). It can be clearly seen from this figures that as more and more agents stop functioning due to battery depletion, the improvement of the total collected information value decreases. We can also observe that by adding node failures into the system, the performance of the algorithms significantly drops down. In particular, in the case of dynamic topology with $\lambda = 0.9$ (Figure 7.1b), the performance of the algorithms is decreased by more than 50%, compared to the case of static topology (Figure 7.1a). One reason would be the fact that, as the environment becomes more dynamic, the budget–limited MAB algorithms cannot efficiently follow the change of the environment, and thus, they produce poor performance.

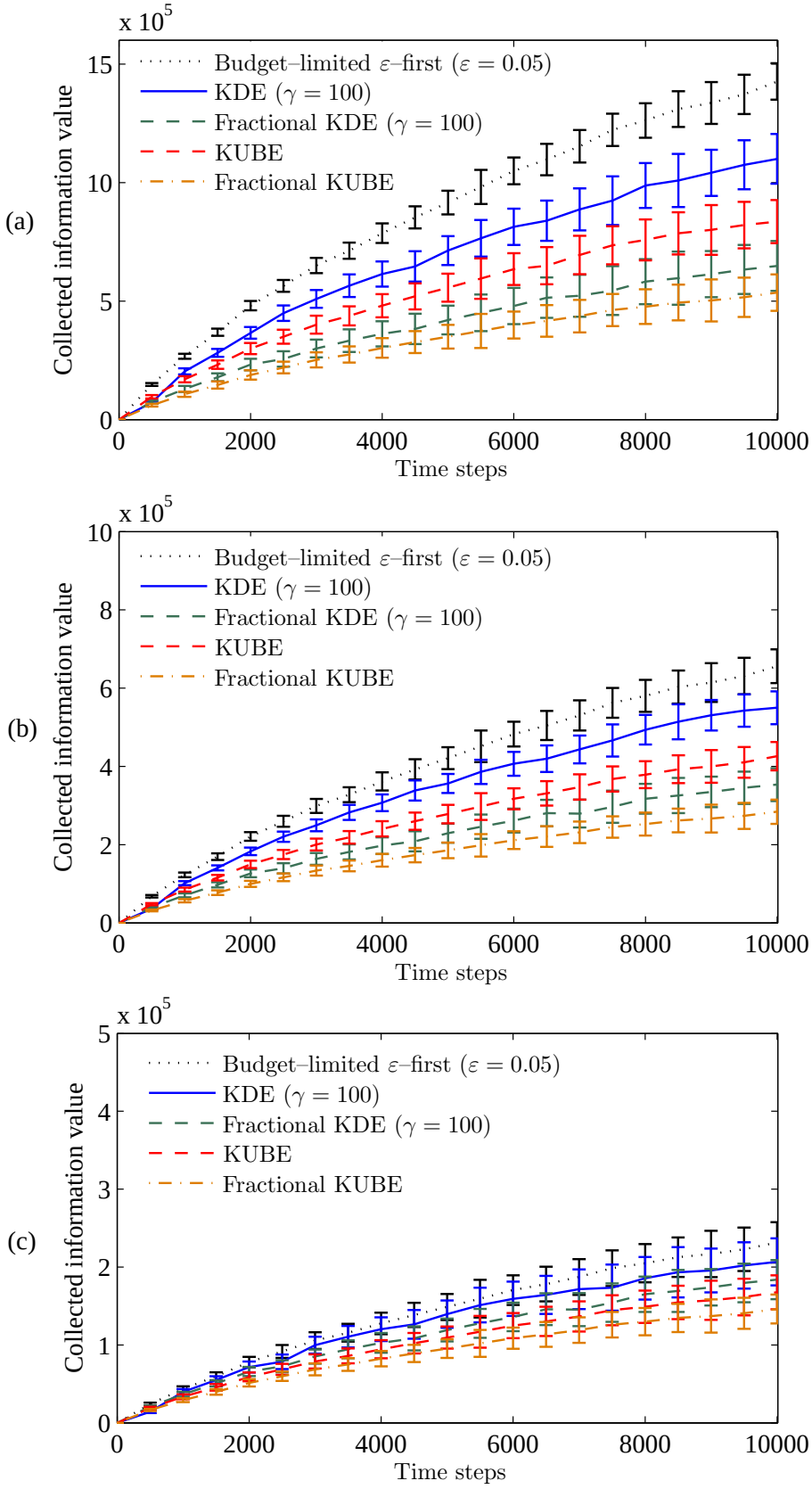


FIGURE 7.1: Information collection in a 100–agent wireless sensor network with (A) static topology with $\lambda = 0.9$; (B) dynamic topology with $\lambda = 0.9$; and (C) dynamic topology with $\lambda = 0.5$.

Note that by modifying the value of λ to be 0.5, the performance of the algorithms is decreased even more. This is due to the fact that in this case, MITRA provides poor performance, since it is not designed for rapid data delivery. We will discuss this issue in more detail in the next section.

7.5.3 Performance Comparison with USAC

Having analysed the performance of MAB/EM using different budget–limited MAB algorithms, we now compare its performance with other state–of–the–art information collecting algorithms within the domain of WSNs. Similar to the previous section, here we also combine MAB/EM with MITRA_8 (i.e. $\tau = 8$). Recall that we use the following benchmark algorithms: (i) USAC; (ii) MITRA without MAB/EM; and (iii) a centralised algorithm with perfect knowledge. The first algorithm represents a state–of–the–art approach within the domain of information collection in WSNs. The second algorithm measures the performance of a non–learning algorithm. Finally, the third algorithm provides a theoretical upper bound.

For the sake of simplicity, we assume that at each time step, USAC can perfectly detect each agent's neighbours (i.e. within our simulation, USAC does not have to deal with topology detection). Note that USAC can intelligently allocate each agent's budget to the tasks it thinks are most important (see Padhy *et al.* (2010) for more details). This behaviour makes USAC similar to our approach, and thus, is one of the reasons we choose USAC as a benchmark algorithm.

The empirical results are depicted in Figure 7.2. Apart from the benchmark algorithms, we also depict the performance of MAB/EM using the budget–limited ε–first with $\varepsilon = 0.05$, and MAB/EM with KDE where $\gamma = 100$. The reason of these choices is that they typically outperform the other budget–limited MAB algorithms (see Table 7.1 for more details). Similar to the empirical evaluation within the previous section, the performance of the algorithms are measured in networks with: (i) static topology and $\lambda = 0.9$ (Figure 7.2a); (ii) dynamic topology and $\lambda = 0.9$ (Figure 7.2b); and (iii) dynamic topology and $\lambda = 0.5$ (Figure 7.2c). As we can see from the figures, MAB/EM approaches, in conjunction with MITRA_8, can achieve up to 70% of the performance of the centralised algorithm within the first scenario (i.e. networks with static topology). However, as the nodes failures are taken into account, this ratio is significantly decreased (see Figures 7.2b and 7.2c). Note that the centralised algorithm becomes computationally infeasible after a certain point. In contrast, our approaches requires low computational complexity, and thus, are computationally feasible.

In addition, note that MITRA with a fixed budget only achieves 50% of the performance of the MAB/EM, illustrating that MITRA itself cannot efficiently collect the information

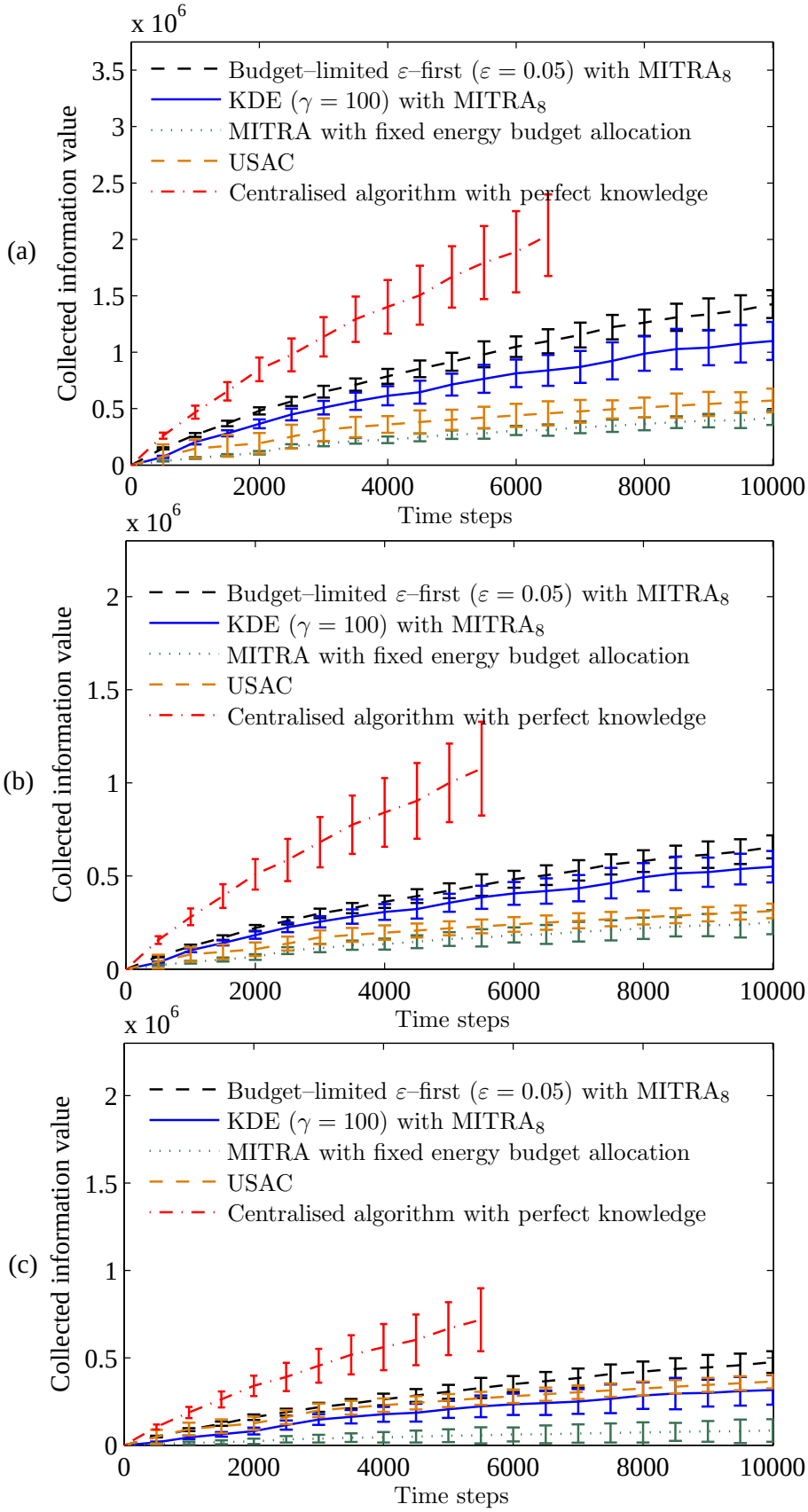


FIGURE 7.2: Performance comparison with USAC in a 100–agent wireless sensor network with (A) static topology with $\lambda = 0.9$; (B) dynamic topology with $\lambda = 0.9$; and (C) dynamic topology with $\lambda = 0.5$.

from the environment, compared to state–of–the–art algorithms, such as USAC. Rather, it must be combined with an adaptive method to allocate the energy budget.

It can also be observed that both the budget–limited ε–first and KDE, in conjunction with MITRA$_8$, outperform USAC by up to 90% in non real–time systems (see Figures 7.2a and 7.2b). However, within real–time systems, where the information value of the packets rapidly decreases over time, our approaches do not outperform USAC. Instead, MITRA shows a significant decrease in terms of performance, that also affects on the performance of the budget–limited ε–first with MITRA$_8$, and KDE with MITRA$_8$, respectively. The reason here is that the routing phase of USAC can guarantee the delivery of packets towards the BS within a time threshold by choosing a full routing path (see Padhy *et al.* (2010) for more details). In contrast, such guarantees do not hold within MITRA. Therefore, within MITRA, a large portion of collected packets are delayed within the network, and thus, their information value is typically close to 0 when the BS receives them. As a result, we can conclude that within real–time systems, MAB/EM in conjunction with MITRA$_8$ cannot outperform USAC, since they are not designed for such systems.

In summary, we can say that by combining MAB/EM with MITRA$_8$, we can outperform state–of–the–art algorithms, such as USAC in systems with low information durability factor. In addition, we also demonstrated that without efficient energy management, MITRA cannot achieve efficient performance, compared to that of USAC.

7.5.4 Performance Evaluation of MITRA$_\tau$

Given the simulation results in the previous section, we can see that MITRA$_\tau$, together with MAB/EM, performs well with $\tau = 8$. As mentioned in Secton 7.4.4, the advantage of using MITRA$_\tau$ instead of MITRA is that the former has limited communication cost. This limitation implies that the performance of MITRA$_\tau$ is decreased, compared to that of MITRA, which is proveably optimal. However, we shall now show that MITRA$_\tau$ still achieves near–optimal performance, even with small values of τ, by studying the performance of MITRA$_\tau$ with different values of τ. The performance of these MITRA$_\tau$ algorithms is compared to that of MITRA with an unlimited number of communication rounds. Note that MITRA may use tens of rounds in order to achieve optimal routing performance (as outlined in Section 7.4.4).

Given this, the numerical results are depicted in Figure 7.3. In particular, the figure shows the performance of MITRA$_\tau$ with $\tau = \{1, 2, 4, 8\}$ [2]. From Figure 7.3, we can see that MITRA$_1$ achieves the lowest performance (it performs 60% less well than the optimal solution in the case of networks with 100 agents). With $\tau = 2$, and $\tau = 4$,

[2]Note that we also have evaluated the performance of MITRA$_\tau$ with higher values of τ, but their improvement is not significant, compared to that of MITRA$_8$.

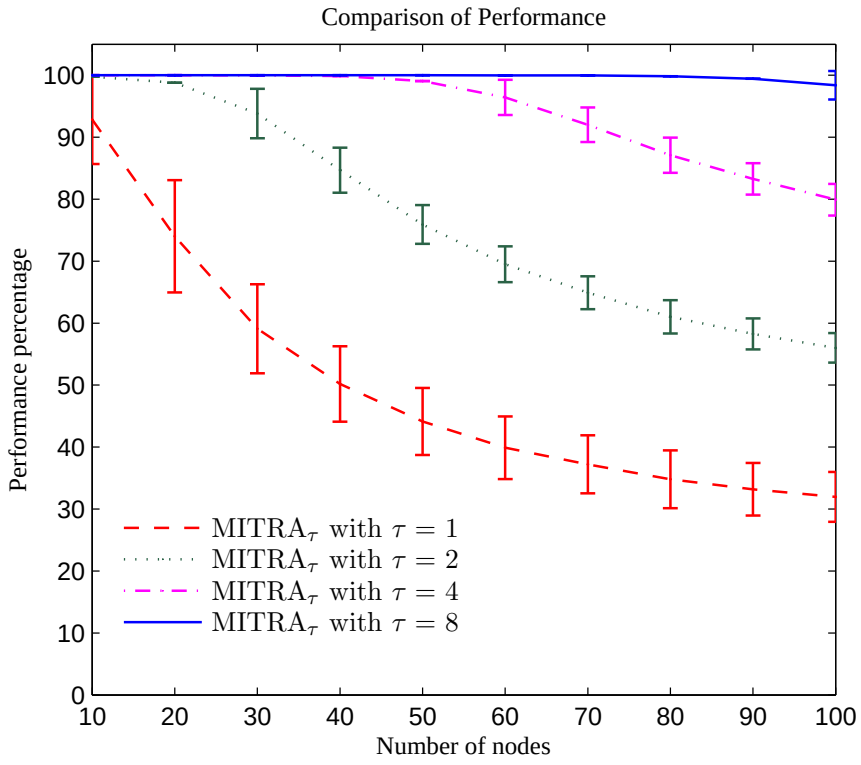


FIGURE 7.3: Performance comparison of MITRA$_\tau$ with that of the unlimited MITRA. The optimal performance achieved by MITRA is 100%.

MITRA$_\tau$ achieves better results, but their performance loss (i.e. the difference between their performance and that of the optimal solution) is still significant. In particular, MITRA$_2$ performs, on average, 60%, whilst MITRA$_4$ achieves around 80% of the optimal solution in the case of 100 agents. In contrast, we can see that with $\tau = 8$, even in the case of networks with 100 agents, the performance of MITRA$_\tau$ is around 98% of the optimal unlimited MITRA. That is, by limiting the number of communication rounds that MITRA can use to $\tau = 8$, our approach still achieves near–optimal solution with around 2% performance loss. On the other hand, according to Theorem 7.5, MITRA without a communication round limit may use up to 66 rounds in order to achieve optimal routing performance. That is, by limiting the communication rounds to $\tau = 8$, we can reduce the number of communication rounds by 87.5%. Given this, by using MITRA$_8$, the number of used communication rounds is small enough so that the total time needed for coordination will not exceed the size of the time slot. This, in particular, justifies our choice of MITRA$_8$ in Section 7.5.2.

7.6 Summary

In contrast to the previous chapters, where the focus was on the theoretical analysis of the proposed budget–limited MAB algorithms, within this chapter, we studied the empirical efficiency of the algorithms. In so doing, we introduced an application scenario for the budget–limited bandits; the problem of long–term information collection in WSNs. Since this problem is a key research challenge within the domain of WSNs, we also aimed to investigate whether our approaches would outperform the state–of–the–art.

Against this background, we first described the literature of relevance of information collection within WSNs. In particular, we considered the related work from the domains of data sampling, information content valuation, data routing, and energy management. We demonstrated that these methods, especially the routing and energy management algorithms, typically fail to fulfil our research requirements given in Section 1.2, specifically: (i) adaptivity; (ii) robustness and flexibility; and (iii) limited use of communication. Given this, we focused on advancing the state–of–the–art from the routing and energy management perspectives. This yielded in the formalisation of the long–term information collection problem. This problem was later decomposed into two sub–problems: (i) energy management; and (ii) maximal information throughput routing.

After formalising both sub–problems, we used the budget–limited MAB approach to tackle the former. In particular, we first transformed the energy management problem, that an agent node has to face, into a budget–limited bandit model. In so doing, we defined the arms and the corresponding pulling costs. We also defined a reward function, and we proved that by maximising the total reward over time, the agents together maximise the total amount of collected information that are also delivered to the BS (Theorem 7.3). Thus, by using the proposed budget–limited MAB algorithms, we tackled this bandit model. We denoted this bandit based energy management approach as MAB/EM.

For the maximal information throughput routing problem, we devised two decentralised routing algorithms, MITRA (for maximal information throughput routing algorithm), and MITRA$_\tau$, respectively. We proved that MITRA provides the optimal solution for the maximal information throughput routing problem (Theorem 7.4). Furthermore, we also provided an upper bound for the number of communication rounds that MITRA needs to use within a time slot (Theorem 7.5). Since the total number of communication rounds that MITRA uses may be large, we modified MITRA so that the number of communication rounds is reduced. The modification resulted in the introduction of MITRA$_\tau$.

Next, we empirically evaluated the performance of MAB/EM together with MITRA$_\tau$. In particular, we first showed that, among the different budget–limited MAB algorithms,

the budget–limited ε–first achieves the best performance in terms of fulfilling Requirement 1 (i.e. experimental performance quality), while KUBE and fractional KUBE performs the worst. We also demonstrated that by using MAB/EM in conjunction with MITRA_8 in non real–time systems, we could outperform USAC, a state–of–the–art information collecting algorithm. However, we also showed that as the information durability factor is decreased (i.e. real–time requirements have to be guaranteed), the performance of our approaches decreases. In addition, we also empirically showed that by choosing small values of τ, near–optimal routing performance can still be achieved, whilst the number of communication rounds in MITRA is significantly reduced. Given this, the integrated model and the proposed algorithms are particularly useful for non real–time monitoring systems (i.e. the information durability factor is high), in which the environment has to be monitored over a prolonged time interval, and unpredicted, important events should be distinguished from the other events.

Chapter 8

Conclusions

In this chapter, we present a global view on the contributions of this thesis towards the research aim of budget–limited multi–armed bandits. To begin, in Section 8.1, we first summarise the research carried out within each chapter in order to achieve this goal. In so doing, we also explain how we satisfied each of the research requirements that we initially set out at the beginning of this report. Then, in Section 8.2, we outline some general areas of future work that follow from this thesis.

8.1 Summary of Results

Multi–armed bandits are becoming an important tool for intelligent agents faced with the challenge of making decisions under uncertainty, as they present one of the clearest examples of the trade–off between exploration and exploitation. Whilst the standard bandit model does not consider pulling costs, there is an increasing need, driven by real–world applications (e.g. costly medical treatments or the shortest driving path scenario), to develop bandit models that take pulling costs into account. To date, bandit models with such cost constraints typically focus on the case when only the arm pulling within the exploration phase is costly, and is limited by a budget, while arm pulling within the exploration phase is cost–free. However, in many other real–world scenarios, it is not only the exploration phase, but also the exploitation phase, that is limited by a cost budget (e.g. wireless sensors or online advertising).

To address this limitation, we introduced a new bandit model, the budget–limited MAB, in which pulling an arm is costly in both the exploration and exploitation phases, and crucially is limited by a single common budget. As a result, the central problem we addressed in this thesis is to design arm pulling algorithms that efficiently tackle this bandit model. In so doing, we first defined the research requirements, that a pulling

algorithm has to fulfil to achieve high performance. These requirements are: (i) efficient experimental performance quality (Requirement 1); (ii) computational feasibility (Requirement 2); and (iii) efficient finite–time regret bound (Requirement 3). We then formalised the budget–limited bandit model, and we defined its objective to maximise the expected value of the total pay–off.

In light of the aforementioned research requirements, we developed a number of pulling algorithms. First, in Chapter 4, we proposed the budget–limited ε–first algorithm. Next, we developed two UCB based algorithms in Chapter 5, namely: (i) KUBE; and (ii) fractional KUBE. We then introduced two decreasing ε–greedy based algorithms, KDE and fractional KDE, in Chapter 6. The budget–limited ε–first algorithm is an empirically and computationally efficient algorithm, which, however, does not satisfy the theoretical requirement (i.e. Requirement 3). In contrast, UCB based algorithms efficiently satisfy the theoretical requirement, but they fail to produce good empirical performance. Finally, the decreasing ε–greedy based algorithms form a trade–off between theoretical and empirical requirements.

In more detail, in Chapter 4, we first described the budget–limited ε–first algorithm. We then provided a linear regret bound for this algorithm. This bound does not guarantee the fulfilment of Requirement 3. However, by analysing the problem from a PAC manner, we improved the regret bound to be $O\left(B^{\frac{2}{3}}\right)$. That is, the budget–limited ε–first algorithm can only fulfil Requirement 3 with a certain (but high) probability. Computation–wise, we showed that the computational complexity of the budget–limited ε–first is a linear function of B and ε. That is, the algorithm satisfies Requirement 2.

We started Chapter 5 by describing KUBE and its fractional counterpart in more detail. We provided regret bounds for these algorithms, both are logarithmic functions of the budget size. Following this, we proved that these bounds are asymptotically optimal; that is, they only differ from the best possible with a constant factor. Thus, they satisfy Requirement 3. Similar to the case of the budget–limited ε–first, we also showed that KUBE and fractional KUBE have efficient computational cost, and thus, they both fulfil Requirement 2.

We continued with Chapter 6 in which we described KDE and fractional KDE. We proved that, similar to the UCB based algorithms, these algorithms also provide asymptotically optimal regret bounds. Hence, they are both efficient in the fulfilment of Requirement 3. In addition, we also studied the computational complexity of KDE and its fractional counterpart, which were shown to be as efficient as that of the UCB based algorithms. Thus, both KDE and fractional KDE satisfy Requirement 2.

In order to measure the fulfilment towards Requirement 1 (i.e. empirical performance) of the proposed algorithms, we implemented these algorithms in the domain of wireless

sensor networks in Chapter 7. In particular, we tackled the problem of long–term information collection in WSNs. Since this problem is one of the key research challenges within the WSN domain, we further considered three additional research requirements: (i) adaptivity (Requirement 4); (ii) robustness and flexibility (Requirement 5); and (iii) limited use of communication (Requirement 6). To tackle the long–term information collection problem, we first introduced its formal description and then decomposed it into two sub–problems, namely energy management and maximal information throughput routing.

Against this background, we proposed a budget–limited multi–armed bandit based approach called MAB/EM for the energy management problem. In particular, we reduced the energy management problem to a MAB problem, by defining the arms, the costs, and the reward functions for the agents. Thus, by using our proposed budget–limited MAB algorithms, we could efficiently tackle the energy management problem. We also showed that MAB/EM efficiently fulfils Requirements 4, 5, and 6, respectively.

For the maximal information throughput routing problem, we devised two decentralised routing algorithms, MITRA (for maximal information throughput routing algorithm), and MITRA$_\tau$, respectively. We proved that MITRA provides the optimal solution for the maximal information throughput routing problem. Furthermore, we also provided an upper bound for the number of communication rounds that MITRA needs to use within a time slot. Although MITRA can efficiently satisfy Requirements 4 and 5, in some cases it may fail to achieve good performance in terms of fulfiling Requirement 6. In more detail, the total number of communication rounds that MITRA uses may be a large number. As a result, we modified MITRA so that the number of communication rounds is reduced. The modification resulted in the introduction of MITRA$_\tau$. Thus, MITRA$_\tau$ satisfies Requirement 6.

Next, by empirical evaluation, we measured the efficiency of the proposed budget–limited MAB algorithms in terms of fulfiling Requirement 1 (i.e. efficient empirical performance). As a result, we demonstrated that the budget–limited ε–first algorithm significantly outperforms the others, while KUBE and its fractional counterpart show the worst performance. This verified our hypothesis that the decreasing ε–greedy methods (i.e. KDE and fractional KDE) efficiently trade off between theoretical and empirical research requirements.

Following this, we demonstrated the efficiency of MAB/EM in conjunction with MITRA$_\tau$ ($\tau = 8$) against the state–of–the–art information collecting algorithms in the domain of WSNs. In particular, to measure the efficiency of our approach, we compared its performance with that of USAC, a state–of–the–art information collecting algorithm within the domain of WSNs. Moreover, to measure the performance surplus that MAB/EM adds to our approach, we also used a non–learning algorithm, that solely uses MITRA,

as a benchmark method. Both comparisons showed that MAB/EM with MITRA$_\tau$ together are efficient in terms of long–term information collection, since it can adapt to the environmental changes. In particular, we demonstrated that, within systems with high values of information durability factor, our approach outperforms USAC. However, we also showed that as the durability factor is decreased, the performance of our approach also decreases. In addition, we showed that by choosing small values of τ, near–optimal routing performance can still be achieved, whilst the number of communication rounds is significantly reduced. Given this, the integrated model and the proposed algorithms are particularly useful for non–real time monitoring systems (i.e. the information durability factor is high), in which the environment has to be monitored over a prolonged time interval, and unpredicted, important events should be distinguished from the other events.

Thus, when taken together, the contributions presented in this thesis represent a significant advance in the state–of–the–art of both budget–limited multi–armed bandits and long–term information collection in wireless sensor networks. Despite these advances, however, many open problems remain. Given this, in the following section, we examine a number of promising directions for future research.

8.2 Future Work

As we demonstrated in Chapter 7, budget–limited MAB algorithms perform well when the network topology is static. However, as node failure occurs within the network, their performance is significantly decreased. Indeed, all of the proposed MAB algorithms assume that the reward values are stationary, and thus, they cannot currently deal with dynamic environments, where the stationarity of the reward values does not hold. To this end, one immediate area of further research is the development of pulling algorithms that take non–stationarity into consideration. Within this direction, we identify three specific lines of investigation to extend the scope of our work:

- **Piece–wise stationary rewards:** One direct extension of our bandit model is to assume that the reward distributions are stationary within certain time intervals, but not in the whole operating time of the agent. Following the work of DaCosta *et al.* (2008) and Hartland *et al.* (2006), a possible solution would be to detect the change points of the environment, and reset the MAB algorithm at those change points. However, the performance analysis of this approach is not obvious, since the performance of the algorithm also depends on the correctness of the change detection. Given this, the key challenge here is to combine the performance analysis of change detection and that of the budget–limited MAB algorithms to provide efficient regret bounds.

- **Well–behaved changes of rewards:** Apart from piece–wise stationarity, another way to extend our model is to add some assumptions on the change of the reward distributions over time, such that the change itself can be defined by some well–behaved properties. For example, one typical assumption is that reward values that are sampled close to each other in time are chosen from similar distributions; that is, the change of the environment is a rather slow and smooth process. Another assumption is to have converging reward distributions over time. In both cases, instead of reseting the MAB algorithm, we can still use the some of the estimated values from the past to approximate the best current combination of arms. Given this, the challenge here is to exploit the behaviour of the change so that efficient performance analysis can be carried out.

- **Adversarial rewards:** Within this extension, nature can be regarded as an opponent of the agent, and thus, whenever an arm is pulled, the reward value is not randomly chosen from a distribution, but is deterministically provided by nature (Auer *et al.*, 2003). Within this setting, concentration inequalities, such as Chernoff–Hoeffding or Bernstein, cannot be used to analyse the theoretical performance. As a result, new techniques are needed in order to efficiently tackle the adversarial budget–limited MAB.

Other possible extensions can be achieved by combining other MAB variants with the budget–limited bandit model (see Section 2.3 for more details). In particular, we aim to address the problems of budget–limited bandits with: (i) side information; and (ii) continnum arms. It is easy to see that the algorithms proposed within this thesis are not suitable to solve these problems. This implies that new techniques are needed to be developed.

Apart from this, we also aim to extend our focus to the more general models of decision making under uncertainty, such as Markov decision processes (MDP) (Sutton and Barto, 1998), or partially observable MDPs (PoMDP) (Cassandra, 1998). The former can be regarded as an extension of the bandit model, since it allows the agent to modify the state of the system by pulling an arm (the MAB model can described as a one–state, or stateless, MDP). The latter is an MDP in which some of the information (e.g. state change of the system or the received reward value) is not avialable to the agent. The main challenge within these models is that by modifying the state of the system, the set of available arms, and thus, the corresponding pulling cost, may change as well. Recall that the underlying knapsack based approach of the budget–limited MAB relies on the fact that the set of arms, and thus, the pulling costs are fixed over time. This implies that the knapsack based techniques, which form the basis of our approaches within this thesis, do not fit to these extensions. This makes the extensions more complex, and thus, non–trivial.

By meeting these challenges, the results related to the budget–limited multi–armed bandits developed in this thesis can be further increased, which will bring a wider applicability of the budget–limited bandit model in many real–world applications.

Bibliography

R. Agrawal. The continuum–armed bandit problem. *SIAM Journal on Control and Optimization*, 33:1926–1951, 1995a.

R. Agrawal. Sample mean based index policies with o(log n) regret for the multi-armed bandit problem. *Advances in Applied Probability*, 27:1054–1078, 1995b.

R. Agrawal, M. Hegde, and D. Teneketzis. Asymptotically efficient allocations rules for multi–armed bandit problem with switching cost. *In the Proceeding of the Twnety–Sixth IEEE Transactions on Automatic Control*, AC–32:968–982, 1988.

F. Ahdi, V. Srinivasan, and K.-C. Chua. Topology control for delay sensitive applications in wireless sensor networks. *Mobile Networks and Applications*, 12(5):406–421, 2007.

K. Akkaya and M. Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325–349, 2005.

I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38:393–422, 2002.

J. N. Al-Karaki and A. E. Kamal. Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications*, 11(6):6–28, 2004.

C. Allenberg, P. Auer, L. Györfi, and Gy. Ottucsák. Hannan consistency in on-line learning in case of unbounded losses under partial monitoring. *Lecture Notes in Computer Science*, 4264:229–243, 2006.

V. Anantharam, P. Varaiya, and J. Walrand. Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays-part i: I.i.d. rewards. *IEEE Transactions on Aumatic Control*, 32(11):977–982, 1987.

G. Anastasi, M. Conti, A. Falchi, E. Gregori, and A. Passarella. Performance measurements of mote sensor networks. *Proceedings of the ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile System*, pages 174–181, 2004.

C. M. Anderson. *Behaviorial Models of Strategies in Multi–Armed Bandit Problems.* PhD thesis, California Institute of Technology, Pasadena, California USA, 2001.

R. Andonov, V. Poirriez, and S Rajopadhye. Unbounded knapsack problem: dynamic programming revisited. *European Journal of Operational Research*, 123(2):394–407, 2000.

A. Antos, V. Grover, and Cs. Szepesvári. Active learning in multi-armed bandits. *In Proceedings of the Nineteenth International Conference on Algorithmic Learning Theory*, pages 287–302, 2008.

J-Y. Audibert, R. Munos, and Cs. Szepesvári. Exploration-exploitation trade-off using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410:1876–1902, 2009.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite–time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *The Society for Industrial and Applied Mathematics Journal on Computing*, 32(1):48–77, 2003.

P. Auer and R. Ortner. Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1–2):55–65, 2010.

P. Auer, R. Ortner, and Cs. Szepesvári. Improved rates for the stochastic continuum–armed bandit problem. *In the Proceedings of the Twentieth Conference on Learning Theory*, pages 454–468, 2007.

D. A. Babayev and S. Mardanov. Reducing the number of variables in integer and linear programming problems. *Computational Optimization and Applications*, 3:99–109, 1994.

Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. Wiley Interscience, 2001. ISBN 047141655X.

R. E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

A. Beygelzimer, J. Langford, L. Li, L. Reyzin, and R.E. Schapire. Contextual bandit algorithms with supervised learning guarantees. *In the Proceeding of the Forteenth International Conference on Artificial Intelligence and Statistics*, 2011.

A. Boukerche. *Algorithms and Protocols for Wireless Sensor Networks*. WileyBlackwell, 2008. ISBN : 0471798134.

D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. *In Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications*, pages 22–31, 2002.

S. Bubeck. *Bandits Games and Clustering Foundations*. PhD thesis, Universit Lille 1, Lille, France, 2010.

S. Bubeck, R. Munos, and G. Stoltz. Pure exploration for multi-armed bandit problems. *In Proceedings of the Twentieth international conference on Algorithmic Learning Theory*, pages 23–37, 2009.

S. Bubeck, R. Munos, G. Stoltz, and Cs. Szepesvári. X-armed bandits. *Journal of Machine Learning Research*, 12:1587–1627, 2011.

A. R. Cassandra. *Exact and Approximate Algorithms for Partially Observable Markov Decision Processes*. PhD thesis, Brown University, Department of Computer Science, Providence, RI, USA, 1998.

N. Cesa-Bianchi and P. Fischer. Finite-time regret bounds for the multiarmed bandit problem. *In the Proceedings of the Fifteenth International Conference on Machine Learning*, pages 100–108, 1998.

N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

N. Cesa-Bianchi and G. Lugosi. Combinatorial bandits. *In the Proceedings of the Twenty-Second Annual Conference on Learning Theory*, pages 1–34, 2009.

N. Cesa-Bianchi, G. Lugosi, and G. Stoltz. Minimizing regret with label efficient prediction. *IEEE: Transactions on Information Theory*, 51:2152–2162, 2005.

D. Chakrabarti, R. Kumar, F. Radlinski, and E. Upfal. Mortal multi-armed bandits. *In Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*, pages 273–280, 2008.

C.-Y. Chong and S. P. Kumar. Sensor networks: Evolution, opportunities and challenges. *Proceedings of IEEE*, 91(8):1247–1256, 2003.

M. Chu, H. Haussecker, and F. Zhao. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *International Journal of High Performance Computing Applications*, 16(3):293–313, 2002.

V. A. Cicirello and S. F. Smith. The max k–armed bandit: a new model of exploration applied to search heuristic selection. *In Proceedings of the Nineteenth Conference on Artificial Intelligence*, pages 1355–1361, 2005.

M. K. Clayton. Covariate models for bernoulli bandits. *Sequential Analysis*, 8(4):405–426, 1989.

E. Cope. Regret and convergence bounds for immediate–reward reinforcement learning with continuous action spaces. *IEEE Transactions on Automatic Control*, 54(6):1243–1253, 2009.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, editors. *Introduction to Algorithms (second edition).* MIT Press and McGraw-Hill, 2001. ISBN 0262032937.

T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley and Sons, 2006. ISBN 0471241954.

L. DaCosta, A. Fialho, M. Schoenauer, and M. Sebag. Adaptive operator selection with dynamic multi–armed bandits. *In Proceedings of the Tenth Annual Conference on Genetic and Evolutionary Computation*, pages 913–920, 2008.

G. B. Dantzig. Discrete variable extremum problems. *Operations Research*, 5:266–277, 1957.

J. Degesys and R. Nagpal. Towards desynchronization of multi-hop topologies. *In Proceedings of the 2008 Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, pages 129–138, 2008.

A. Dekorsy, J. Fliege, and M. S´ollner. Optimal distributed routing and power control decomposition for wireless networks. *In Proceedings of the Fiftieth IEEE Global Telecommunications Conference*, pages 4920–4924, 2007.

I. Demirkol, C. Ersoy, and F. Alagoz. Mac protocols for wireless sensor networks: a survey. *IEEE Communications Magazine*, 44(4):115–121, 2006.

W. Dinga, S.S. Iyengara, R. Kannana, and W. Rummler. Energy equivalence routing in wireless sensor networks. *Microprocessors and Microsystems*, 28:467–475, 2004.

K. Dudzinski. A note on dominance relation in unbounded knapsack problems. *Operations Research Letters*, 10:417–419, 1991.

J. Elson and D. Estrin. Time synchronization forwireless sensor networks. *Proceedings of the 2001 International Parallel and Distributed Processing Symposium (IPDPS), Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, 2001.

E. Even-Dar, S. Mannor, and Y. Mansour. Pac bounds for multi–armed bandit and markov decision processes. *In the Proceedings of the Fifteenth Annual Conference on Computational Learning Theory*, pages 255–270, 2002.

V. F. Farias and R. Madan. The irrevocable multi-armed bandit problem. *Operations Research*, 59(2), 2011.

B. R. Frieden. *Science from Fisher Information: A Unification.* Cambridge University Press, 2004. ISBN 0521009111.

C. Guestrin, A. Krause, and A. P. Singh. Near-optimal sensor placements in gaussian processes. *In Proceedings of the Twenty-second International Conference on Machine Learning*, pages 265–272, 2005.

S. Guha and K. Munagala. Approximation algorithms for budgeted learning problems. *In Proceedings of the Thirty-Ninth Annual ACM symposium on Theory of Computing*, pages 104–113, 2007.

S. Guha and K. Munagala. Multi-armed bandits with metric switching costs. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris Nikoletseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming*, volume 5556 of *Lecture Notes in Computer Science*, pages 496–507. Springer Berlin / Heidelberg, 2009.

J. P. Hardwick and Q. F. Stout. Bandit strategies for ethical sequential allocation. *Computing Science and Statistics*, 23:421–424, 1991.

C. Hartland, S. Gelly, N. Baskiotis, O. Teytaud, and M. Sebag. Multiarmed bandit, dynamic environments and meta–bandits. *Online Trading of Exploration and Exploitation Workshop, NIPS*, 2006.

T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J.A. Stankovic, and T. Abdelzaher. Achieving real-time target tracking using wireless sensor networks. *In Proceedings of the Twelfth IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 37–48, 2006.

W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *In Proceedings of the Hawaii International Conference on System Sciences*, pages 1–10, 2000.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *ournal of the American Statistical Association*, 58:13–30, 1963.

J. Honda and A. Takemura. An asymptotically optimal bandit algorithm for bounded support models. *In the Proceedings of the Twenty–Third Annual Conference on Learning Theory*, pages 67–79, 2010.

I. S. Hwang, K. Roy, H. Balakrishnan, and C. Tomlin. A distributed multiple-target identity management algorithm in sensor networks. *In Proceedings of the Fourty-Third IEEE Conference on Decision and Control*, pages 728–734, 2004.

O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM*, 4:197–204, 1975.

C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions on Networking*, 11(1):2–16, 2003.

A. Ivic, editor. *The Riemann Zeta Function*. John Wiley & Sons, 1985. ISBN 0-471-80634-X.

A. Jain and E. Y. Chang. Adaptive sampling for sensor networks. *In Proceedings of the First Workshop on Data Management for Sensor Networks*, pages 10–16, 2004.

N. R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4):35–41, 2001.

L. P. Kaelbling, editor. *Learning in embedded systems*. MIT Press, 1993.

A. Kansal and M. B Srivastava. An environmental energy harvesting framework for sensor networks. *In Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, pages 481–486, 2003.

H. Kellerer and U. Pferschy. A new fully polynomial time approximation scheme for the knapsack problem. *Journal of Combinatorial Optimization*, 3:59–71, 1999.

H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 2004.

J. Kho, Long L. Tran-Thanh, A. Rogers, and N. R. Jennings. An agent-based distributed coordination mechanism for wireless visual sensor nodes using dynamic programming. *The Computer Journal*, 53(8):1277–1290, 2010.

J. Kho, A. Rogers, and N. R. Jennings. Decentralised adaptive sampling of wireless sensor networks. *ACM Transactions on Sensor Networks*, 5(3):19–53, 2009.

R. Kleinberg. Nearly tight bounds for the continuum–armed bandit problem. *Advances in Neural Information Processing Systems*, 17:697–704, 2005.

R. Kleinberg, A. Slivkins, and E. Upfal. Multi–armed bandits in metric spaces. *In the Proceedings of the Fortieth ACM Symposium on Theory of Computing*, 2008.

R. Kohli, R. Krishnamurti, and P. Mirchandani. Average performance of greedy heuristics for the integer knapsack problem. *European Journal of Operational Research*, 154(1):36–45, 2004.

A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements:maximizing information while minimizing communication cost. *In Proceedings of the Fifth International Conference on Information Processing in Sensor Networks*, pages 2–10, 2006.

V. Kuleshov and D. Precup. Algorithms for the multi-armed bandit problem. *Unpublished*, 2010.

T. L. Lai and T. W. Lim. Optimal stopping for brownian motion with applications to sequential analysis and option pricing. *Journal of Statistical Planning and Inference*, 130(1-2):21–47, 2005.

T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathemathics*, 6(1):4–22, 1985.

J. Langford and T. Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. *In Proceedings of the Twenty–First Annual Conference on Neural Information Processing Systems*, 2007.

E. L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4:339–356, 1979.

V. Lesser, C. Ortiz, and M. Tambe, editors. *Distributed sensor networks: a multiagent perspective*. Kluwer Publishing, 2003. ISBN 1402074999.

P. Li, Y. Gu, and B. Zhao. A global-energy-balancing real-time routing in wireless sensor networks. *In Proceedings of the IEEE Asia-Pacific Services Computing Conference*, pages 89–93, 2007.

S. Lindsey and C. S. Raghavendra. Pegasis: Power efficient gathering in sensor information systems. *In Proceedings of the IEEE Aerospace Conference*, 3:3.1125–3.1130, 2002.

T. Lu, D. Pál, and M. Pál. Contextual multi–armed bandits. *In the Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 485–492, 2010.

R. D. Luce, editor. *Individual choice behavior*. Wiley New York, 1959.

O. Madani, D. J. Lizotte, and R. Greiner. The budgeted multi–armed bandit problem. *In Proceedings of the Seventeenth Annual Conference on Learning Theory*, pages 643–645, 2004.

O.-A. Maillard, R. Munos, and G. Stoltz. A finite-time analysis of multi-armed bandits problems with kullback-leibler divergences. *In the Proceedings of the Twenty–Fourth Annual Conference on Learning Theory*, 2011.

G. Mainland, D. C. Parkes, and M.Welsh. Decentralised, adaptive resource allocation for sensor networks. *In Proceedings of the Second USENIX/ACM Symposium on Networked Systems Design and Implementation*, pages 315–328, 2005.

S. Marcello and M. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. Wiley, 1990.

K. Martinez, J. Hart, and R. Ong. Environmental sensor networks. *Computer*, 37(8): 50–56, 2004.

G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy. Ultra-low power data storage for sensor networks. *Proceedings of the Fifth international conference on Information processing in sensor networks*, pages 374–381, 2006.

G. V. Merrett. *Energy- and Information-Managed Wireless Sensor Networks: Modelling and Simulation*. PhD thesis, University of Southampton, School of Electronics and Computer Science, Southampton UK, 2008.

C. Ok, S. Lee, P. Mitra, and S. Kumara. Distributed energy balanced routing for wireless sensor networks. *Computers & Industrial Engineering*, 57:125–135, 2009.

M. A. Osborne, A. Rogers, S. Ramchurn, S. J. Roberts, and N. R. Jennings. Towards real-time information processing of sensor network data using computationally efficient multi-output gaussian processes. *In Proceedings of the Seventh International Conference on Information Processing in Sensor Networks*, pages 109–120, 2008.

Gy. Ottucsák. *Machine–Learning for Infocommunication Systems*. PhD thesis, Budapest University of Technology and Economics, Budapest, Hungary, 2007.

P. Padhy, R. K. Dash, K. Martinez, and N. R. Jennings. A utility-based adaptive sensing and multihop communication protocol for wireless sensor networks. *ACM Transactions on Sensor Networks*, 6(3):1–39, 2010.

N.G. Pavlidis, D.K. Tasoulis, and D.J. Hand. Simulation studies of multi–armed bandits with covariates. *In Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, pages 493–498, 2008.

M. Pechoucek and V. Marik. Industrial deployment of multi-agent technologies: review and selected case studies. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 17(3):397–431, 2008.

D. Pisinger. Where are the hard knapsack problems? *Computers and Operations Research*, 32(9):2271–2284, 2005.

F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. *In the Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pages 784–791, 2008.

P. Rigollet and A. Zeevi. Nonparametric bandits with covariates. *In the Proceedings of the Twenty–Third Conference on Learning Theory*, pages 54–66, 2010.

H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the AMS*, 55:527–535, 1952.

A. Rogers, D. D. Corkill, and N. R. Jennings. Agent technologies for sensor networks. *IEEE Intelligent Systems*, 24(2):13–17, 2009.

A. Rogers, E. David, and N. R. Jennings. Self-organised routing for wireless microsensor networks. *IEEE Transactions on Systems, Man, and Cybernetics (Part A)*, 35(3):349–359, 2005.

K. Romer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54–61, 2004.

G. Simon, A. Ledeczi, and M. Maroti. Sensor network-based countersniper system. *In Proceedings of the Second International Conference on Embedded Networked Sensor Systems*, pages 1–12, 2004.

S. Singh, M. Woo, and C. S. Raghavendra. Power-aware routing in mobile ad hoc networks. *In Proceedings of the Fourth annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 181–190, 1998.

S. S. Skiena. Who is interested in algorithms and why?: lessons from the stony brook algorithms repository. *SIGACT News*, 30(3):65–74, 1999.

L.-K. Soh and C. Tsatsoulis. A real-time negotiation model and a multi-agent sensor network implementation. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 11(3):215–271, 2005.

J. A. Stankovic. Research challenges for wireless sensor networks. *ACM SIGBED Review*, 1(2):9–12, 2004.

B. Sundararaman, U. Buy, and A. D. Kshemkalyani. Clock synchronization for wireless sensor networks: a survey. *Ad Hoc Networks*, 3(3):281–323, 2005.

R. S. Sutton and A. G. Barto, editors. *Reinforcement Learning: An Introduction*. MIT Press, 1998. ISBN 0-262-19398-1.

A. M. Sykulski. *The Exploration-Exploitation Trade-Off in Sequential Decision Making Problems*. PhD thesis, Imperial College London, London, UK, 2011.

R. Torah, P. Glynne-Jones, M. Tudor, T. O'Donnell, S. Roy, and S. Beeby. Self-powered autonomous wireless sensor node using vibration energy harvesting. *Measurement Science and Technology*, pages 125202.1–125202.8, 2008.

L. Tran-Thanh, A. Chapman, J. E. Munoz de Cote, A. Rogers, and N. R. Jennings. Epsilon–first policies for budget–limited multi–armed bandits. *In Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*, pages 1211–1216, 2010.

L. Tran-Thanh, A. Rogers, and N. R. Jennings. Long–term information collection with energy harvesting wireless sensors: A multiarmed bandit based approach. *Journal of Autonomous Agents and Multi-Agent Systems*, 2011.

L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134 – 1142, 1984.

J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. *European Conference on Machine Learning*, pages 437–448, 2005.

D. Wagner and R. Wattenhofer. *Algorithms for Sensor and Ad Hoc Networks: Advanced Lectures*. Springer, 2007. ISBN : 354074990X.

X. Wang and Y. Wang. Optimal investment and consumption with stochastic dividends. *Applied Stochastic Models in Business and Industry*, page doi:10.1002/asmb.823, 2009.

C. J. C. H. Watkins. *Learning from delayed rewards*. PhD thesis, Cambridge University, Cambridge, UK, 1989.

R. Willett, A. Martin, and R. Nowak. Backcasting: Adaptive sampling for sensor networks. *In Proceedings of the Third International Symposium on Information Processing in Sensor Networks*, pages 124–156, 2004.

P. Wittle. Arm–acquiring bandits. *The Annals of Probability*, 9(2):284–292, 1981.

P. Wittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25A:287–298, 1988.

M. Woodfoofe. Sequential allocation with covariates. *The Indian Journal of Statistics*, 44(3):403–414, 1982.

T. Wu and S. Biswas. Minimizing inter-cluster interference by self-reorganizing mac allocation in sensor networks. *Wireless Networks*, 13(5):691–703, 2007.

Mohamed Younis, Kemal Akkaya, Mohamed Eltoweissy, and Ashraf Wadaa. On handling qos traffic in wireless sensor networks. *Hawaii International Conference on System Sciences*, 9:90292a, 2004.

F. Zhao and L. J. Guibasn. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann, 2004. ISBN 1558609148.

J. Zhou and D. de Roure. Floodnet: Coupling adaptive sampling with energy aware routing in a flood warning system. *Journal of Computer Science and Technology*, 22(1):121–130, 2007.