

Bayesian Reinforcement Learning: A Survey

Mohammad Ghavamzadeh

Adobe Research & INRIA
mohammad.ghavamzadeh@inria.fr

Shie Mannor

Technion
shie@ee.technion.ac.il

Joelle Pineau

McGill University
jpineau@cs.mcgill.ca

Aviv Tamar

University of California, Berkeley
avivt@berkeley.edu

now

the essence of knowledge

Boston — Delft



Documents

Document Sharing Hub

Foundations and Trends[®] in Machine Learning

Published, sold and distributed by:

now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:

now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar. *Bayesian Reinforcement Learning: A Survey*. Foundations and Trends[®] in Machine Learning, vol. 8, no. 5-6, pp. 359–483, 2015.

This Foundations and Trends[®] issue was typeset in L^AT_EX using a class file designed by Neal Parikh. Printed on acid-free paper.

ISBN: 978-1-68083-089-7

© 2015 M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The 'services' for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Foundations and Trends[®] in Machine Learning

Volume 8, Issue 5-6, 2015

Editorial Board

Editor-in-Chief

Michael Jordan

University of California, Berkeley
United States

Editors

Peter Bartlett

UC Berkeley

Yoshua Bengio

University of Montreal

Avrim Blum

CMU

Craig Boutilier

University of Toronto

Stephen Boyd

Stanford University

Carla Brodley

Tufts University

Inderjit Dhillon

UT Austin

Jerome Friedman

Stanford University

Kenji Fukumizu

ISM, Japan

Zoubin Ghahramani

University of Cambridge

David Heckerman

Microsoft Research

Tom Heskes

Radboud University

Geoffrey Hinton

University of Toronto

Aapo Hyvarinen

HIIT, Finland

Leslie Pack Kaelbling

MIT

Michael Kearns

UPenn

Daphne Koller

Stanford University

John Lafferty

University of Chicago

Michael Littman

Brown University

Gabor Lugosi

Pompeu Fabra University

David Madigan

Columbia University

Pascal Massart

University of Paris-Sud

Andrew McCallum

UMass Amherst

Marina Meila

University of Washington

Andrew Moore

CMU

John Platt

Microsoft Research

Luc de Raedt

University of Freiburg

Christian Robert

U Paris-Dauphine

Sunita Sarawagi

IIT Bombay

Robert Schapire

Princeton University

Bernhard Schoelkopf

MPI Tübingen

Richard Sutton

University of Alberta

Larry Wasserman

CMU

Bin Yu

UC Berkeley

Editorial Scope

Topics

Foundations and Trends® in Machine Learning publishes survey and tutorial articles on the theory, algorithms and applications of machine learning, including the following topics:

- Adaptive control and signal processing
- Applications and case studies
- Behavioral, cognitive, and neural learning
- Bayesian learning
- Classification and prediction
- Clustering
- Data mining
- Dimensionality reduction
- Evaluation
- Game theoretic learning
- Graphical models
- Independent component analysis
- Inductive logic programming
- Kernel methods
- Markov chain Monte Carlo
- Model choice
- Nonparametric methods
- Online learning
- Optimization
- Reinforcement learning
- Relational learning
- Robustness
- Spectral methods
- Statistical learning theory
- Variational inference
- Visualization

Information for Librarians

Foundations and Trends® in Machine Learning, 2015, Volume 8, 6 issues. ISSN paper version 1935-8237. ISSN online version 1935-8245. Also available as a combined paper and online subscription.

Bayesian Reinforcement Learning: A Survey

Mohammad Ghavamzadeh
Adobe Research & INRIA
mohammad.ghavamzadeh@inria.fr

Shie Mannor
Technion
shie@ee.technion.ac.il

Joelle Pineau
McGill University
jpineau@cs.mcgill.ca

Aviv Tamar
University of California, Berkeley
avivt@berkeley.edu

Contents

1	Introduction	2
2	Technical Background	9
2.1	Multi-Armed Bandits	9
2.2	Markov Decision Processes	12
2.3	Partially Observable Markov Decision Processes	16
2.4	Reinforcement Learning	18
2.5	Bayesian Learning	20
3	Bayesian Bandits	27
3.1	Classical Results	28
3.2	Bayes-UCB	31
3.3	Thompson Sampling	31
4	Model-based Bayesian Reinforcement Learning	38
4.1	Models and Representations	38
4.2	Exploration/Exploitation Dilemma	42
4.3	Offline Value Approximation	43
4.4	Online near-myopic value approximation	45
4.5	Online Tree Search Approximation	47
4.6	Methods with Exploration Bonus to Achieve PAC Guarantees	53
4.7	Extensions to Unknown Rewards	60

4.8	Extensions to Continuous MDPs	63
4.9	Extensions to Partially Observable MDPs	64
4.10	Extensions to Other Priors and Structured MDPs	67
5	Model-free Bayesian Reinforcement Learning	69
5.1	Value Function Algorithms	69
5.2	Bayesian Policy Gradient	78
5.3	Bayesian Actor-Critic	86
6	Risk-aware Bayesian Reinforcement Learning	90
7	BRL Extensions	97
7.1	PAC-Bayes Model Selection	97
7.2	Bayesian Inverse Reinforcement Learning	98
7.3	Bayesian Multi-agent Reinforcement Learning	100
7.4	Bayesian Multi-Task Reinforcement Learning	100
8	Outlook	103
	Acknowledgements	106
	Appendices	107
A	Index of Symbols	108
B	Discussion on GPTD Assumptions on the Noise Process	111
	References	113

Abstract

Bayesian methods for machine learning have been widely investigated, yielding principled methods for incorporating prior information into inference algorithms. In this survey, we provide an in-depth review of the role of Bayesian methods for the reinforcement learning (RL) paradigm. The major incentives for incorporating Bayesian reasoning in RL are: **1)** it provides an elegant approach to action-selection (exploration/exploitation) as a function of the uncertainty in learning; and **2)** it provides a machinery to incorporate prior knowledge into the algorithms. We first discuss models and methods for Bayesian inference in the simple single-step Bandit model. We then review the extensive recent literature on Bayesian methods for model-based RL, where prior information can be expressed on the parameters of the Markov model. We also present Bayesian methods for model-free RL, where priors are expressed over the value function or policy class. The objective of the paper is to provide a comprehensive survey on Bayesian RL algorithms and their theoretical and empirical properties.

M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar. *Bayesian Reinforcement Learning: A Survey*. Foundations and Trends[®] in Machine Learning, vol. 8, no. 5-6, pp. 359–483, 2015.
DOI: 10.1561/22000000049.

1

Introduction

A large number of problems in science and engineering, from robotics to game playing, tutoring systems, resource management, financial portfolio management, medical treatment design and beyond, can be characterized as sequential decision-making under uncertainty. Many interesting sequential decision-making tasks can be formulated as reinforcement learning (RL) problems [Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998]. In an RL problem, an agent interacts with a dynamic, stochastic, and incompletely known environment, with the goal of finding an action-selection strategy, or *policy*, that optimizes some long-term performance measure.

One of the key features of RL is the focus on learning a control policy to optimize the choice of actions over several time steps. This is usually learned from sequences of data. In contrast to supervised learning methods that deal with independently and identically distributed (i.i.d.) samples from the domain, the RL agent learns from the samples that are collected from the trajectories generated by its sequential interaction with the system. Another important aspect is the effect of the agent's policy on the data collection; different policies naturally yield different distributions of sam-

pled trajectories, and thus, impacting what can be learned from the data.

Traditionally, RL algorithms have been categorized as being either *model-based* or *model-free*. In the former category, the agent uses the collected data to first build a model of the domain's dynamics and then uses this model to optimize its policy. In the latter case, the agent directly learns an optimal (or good) action-selection strategy from the collected data. There is some evidence that the first method provides better results with less data [Atkeson and Santamaria, 1997], and the second method may be more efficient in cases where the solution space (e.g., policy space) exhibits more regularity than the underlying dynamics, though there is some disagreement about this,.

A major challenge in RL is in identifying good data collection strategies, that effectively balance between the need to explore the space of all possible policies, and the desire to focus data collection towards trajectories that yield better outcome (e.g., greater chance of reaching a goal, or minimizing a cost function). This is known as the *exploration-exploitation* tradeoff. This challenge arises in both model-based and model-free RL algorithms.

Bayesian reinforcement learning (BRL) is an approach to RL that leverages methods from Bayesian inference to incorporate information into the learning process. It assumes that the designer of the system can express prior information about the problem in a probabilistic distribution, and that new information can be incorporated using standard rules of Bayesian inference. The information can be encoded and updated using a parametric representation of the system dynamics, in the case of model-based RL, or of the solution space, in the case of model-free RL.

A major advantage of the BRL approach is that it provides a principled way to tackle the exploration-exploitation problem. Indeed, the Bayesian posterior naturally captures the full state of knowledge, subject to the chosen parametric representation, and thus, the agent can select actions that maximize the expected gain with respect to this information state.

Another major advantage of BRL is that it implicitly facilitates regularization. By assuming a prior on the value function, the parameters defining a policy, or the model parameters, we avoid the trap of letting a few data points steer us away from the true parameters. On the other hand, having a prior precludes overly rapid convergence. The role of the prior is therefore to soften the effect of sampling a finite dataset, effectively leading to regularization. We note that regularization in RL has been addressed for the value function [Farahmand et al., 2008b] and for policies [Farahmand et al., 2008a]. A major issue with these regularization schemes is that it is not clear how to select the regularization coefficient. Moreover, it is not clear why an optimal value function (or a policy) should belong to some pre-defined set.

Yet another advantage of adopting a Bayesian view in RL is the principled Bayesian approach for handling parameter uncertainty. Current frequentist approaches for dealing with modelling errors in sequential decision making are either very conservative, or computationally infeasible [Nilim and El Ghaoui, 2005]. By explicitly modelling the distribution over unknown system parameters, Bayesian methods offer a promising approach for solving this difficult problem.

Of course, several challenges arise in applying Bayesian methods to the RL paradigm. First, there is the challenge of selecting the correct representation for expressing prior information in any given domain. Second, defining the decision-making process over the information state is typically computationally more demanding than directly considering the natural state representation. Nonetheless, a large array of models and algorithms have been proposed for the BRL framework, leveraging a variety of structural assumptions and approximations to provide feasible solutions.

The main objective of this paper is to provide a comprehensive survey on BRL algorithms and their theoretical and empirical properties. In Chapter 2, we provide a review of the main mathematical concepts and techniques used throughout this paper. Chapter 3 surveys the Bayesian learning methods for the case of single-step decision-making, using the *bandit* framework. This section serves both as an exposition of the potential of BRL in a simpler setting that is well understood, but is

also of independent interest, as bandits have widespread applications. The main results presented here are of a theoretical nature, outlining known performance bounds for the regret minimization criteria. Chapter 4 reviews existing methods for *model-based* BRL, where the posterior is expressed over parameters of the system dynamics model. Chapter 5 focuses on BRL methods that do not explicitly learn a model of the system, but rather the posterior is expressed over the solution space. Chapter 6 focuses on a particular advantage of BRL in dealing with risk due to parameter-uncertainty, and surveys several approaches for incorporating such risk into the decision-making process. Finally, Chapter 7 discusses various extensions of BRL for special classes of problems (PAC-Bayes model selection, inverse RL, multi-agent RL, and multi-task RL). Figure 1.1 outlines the various BRL approaches covered throughout the paper.

An Example Domain

We present an illustrative domain suitable to be solved using the BRL techniques surveyed in this paper. This running example will be used throughout the paper to elucidate the difference between the various BRL approaches and to clarify various BRL concepts.

Example 1.1 (The Online Shop). In the online shop domain, a retailer aims to maximize profit by sequentially suggesting products to online shopping customers. Formally, the domain is characterized by the following model:

- A set of possible customer states, $\mathcal{X}$. States can represent intrinsic features of the customer such as gender and age, but also dynamic quantities such as the items in his shopping cart, or his willingness to shop;
- A set of possible product suggestions and advertisements, $\mathcal{A}$;
- A probability kernel, P , defined below.

An *episode* in the online shop domain begins at time $t = 0$, when a customer with features $x_0 \in \mathcal{X}$ enters the online shop. Then, a sequential interaction between the customer and the online shop begins,

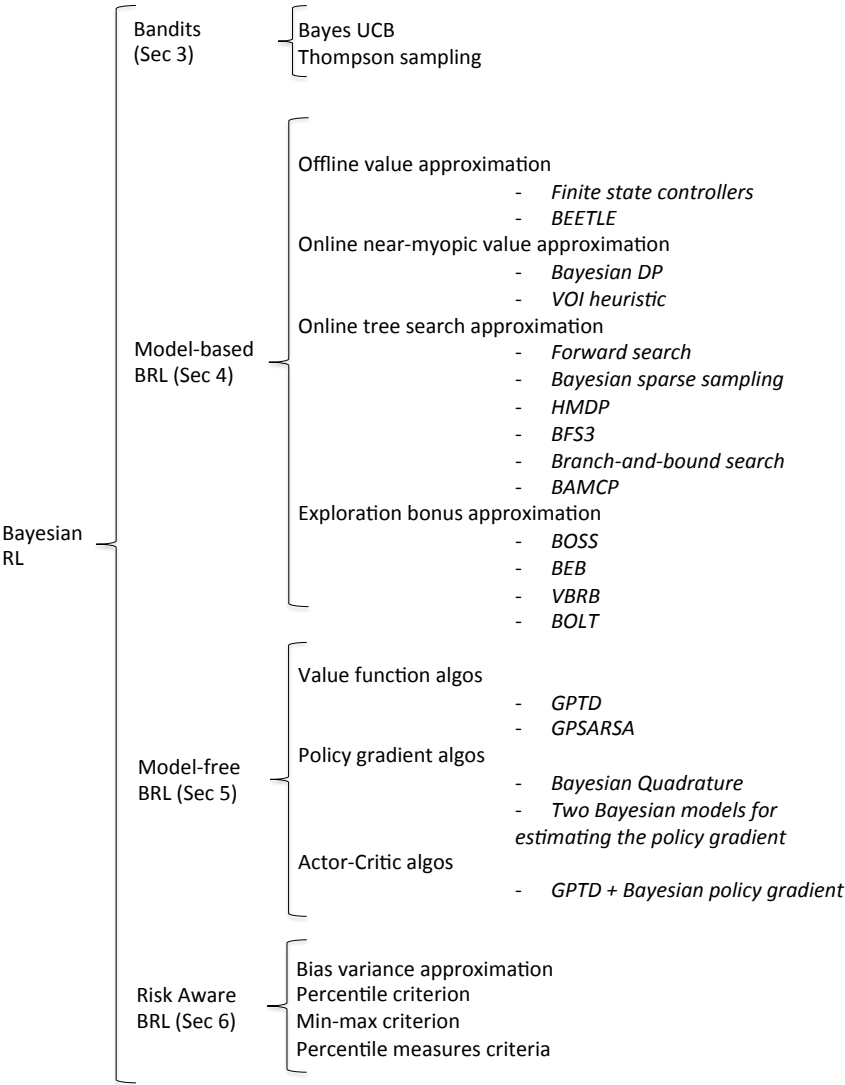


Figure 1.1: Overview of the Bayesian RL approaches covered in this survey.

where at each step $t = 0, 1, 2, \dots$, an advertisement $a_t \in \mathcal{A}$ is shown to the customer, and following that the customer makes a decision to either (i) add a product to his shopping cart; (ii) not buy the product, but continue to shop; (iii) stop shopping and check out. Following the customer's decision, his state changes to x_{t+1} (reflecting the change in the shopping cart, willingness to continue shopping, etc.). We assume that this change is captured by a probability kernel $P(x_{t+1} | x_t, a_t)$.

When the customer decides to check out, the episode ends, and a profit is obtained according to the items he had added to his cart. The goal is to find a product suggestion policy, $x \rightarrow a \in \mathcal{A}$, that maximizes the expected total profit.

When the probabilities of customer responses P are known in advance, calculating an optimal policy for the online shop domain is basically a *planning* problem, which may be solved using traditional methods for resource allocation [Powell, 2011]. A more challenging, but realistic, scenario is when P is not completely known beforehand, but has to be *learned* while interacting with customers. The BRL framework employs *Bayesian* methods for learning P , and for learning an optimal product suggestion policy.

There are several advantages for choosing a Bayesian approach for the online shop domain. First, it is likely that some prior knowledge about P is available. For example, once a customer adds a product of a particular brand to his cart, it is likely that he prefers additional products of the same brand over those of a different one. Taking into account such knowledge is natural in the Bayesian method, by virtue of the *prior* distribution over P . As we shall see, the Bayesian approach also naturally extends to more general forms of *structure* in the problem.

A second advantage concerns what is known as the *exploitation-exploration* dilemma: should the decision-maker display only the most profitable product suggestions according to his current knowledge about P , or rather take exploratory actions that may turn out to be less profitable, but provide useful information for future decisions? The Bayesian method offers a principled approach to dealing with this difficult problem by explicitly quantifying the value

of exploration, made possible by maintaining a *distribution* over P .

The various parameter configurations in the online shop domain lead to the different learning problems surveyed in this paper. In particular:

- For a single-step interaction, i.e., when the episode terminates after a single product suggestion, the problem is captured by the multi-armed bandit model of Chapter 3.
- For small-scale problems, i.e., a small number of products and customer types, P may be learnt explicitly. This is the model-based approach of Chapter 4.
- For large problems, a near-optimal policy may be obtained without representing P explicitly. This is the model-free approach of Chapter 5.
- When the customer state is not fully observed by the decision-maker, we require models that incorporate partial observability; see §2.3 and §4.9.

Throughout the paper, we revisit the online shop domain, and specify explicit configurations that are relevant to the surveyed methods.

2

Technical Background

In this section we provide a brief overview of the main concepts and introduce the notations used throughout the paper. We begin with a quick review of the primary mathematical tools and algorithms leveraged in the latter sections to define BRL models and algorithms.

2.1 Multi-Armed Bandits

As was previously mentioned, a key challenge in sequential decision-making under uncertainty is the exploration/exploitation dilemma: the tradeoff between either taking actions that are most rewarding according to the current state of knowledge, or taking exploratory actions, which may be less immediately rewarding, but may lead to better informed decisions in the future.

The *stochastic multi-armed bandit* (MAB) problem is perhaps the simplest model of the exploration/exploitation tradeoff. Originally formulated as the problem of a gambler choosing between different slot machines in a casino (‘one armed bandits’), the stochastic MAB model features a decision-maker that sequentially chooses actions $a_t \in \mathcal{A}$, and observes random outcomes $Y_t(a_t) \in \mathcal{Y}$ at discrete time steps

$t = 1, 2, 3, \dots$. A known function $r : \mathcal{Y} \rightarrow \mathbb{R}$ associates the random outcome to a reward, which the agent seeks to maximize. The outcomes $\{Y_t(a)\}$ are drawn i.i.d. over time from an unknown probability distribution $P(\cdot|a) \in \mathcal{P}(\mathcal{Y})$, where $\mathcal{P}(\mathcal{Y})$ denotes the set of probability distributions on (Borel) subsets of $\mathcal{Y}$ (see Bubeck and Cesa-Bianchi [2012] for reference).

Model 1 (Stochastic K -Armed Bandit) Define a K -MAB to be a tuple $\langle \mathcal{A}, \mathcal{Y}, P, r \rangle$ where

- $\mathcal{A}$ is the set of actions (arms), and $|\mathcal{A}| = K$,
- $\mathcal{Y}$ is the set of possible outcomes,
- $P(\cdot|a) \in \mathcal{P}(\mathcal{Y})$ is the outcome probability, conditioned on action $a \in \mathcal{A}$ being taken,
- $r(Y) \in \mathbb{R}$ represents the reward obtained when outcome $Y \in \mathcal{Y}$ is observed.

A rule that prescribes to the agent which actions to select, or *policy*, is defined as a mapping from past observations to a distribution over the set of actions. Since the probability distributions are initially unknown, the decision-maker faces a tradeoff between exploitation, i.e., selecting the arm she believes is optimal, or making exploratory actions to gather more information about the true probabilities. Formally, this tradeoff is captured by the notion of *regret*:

Definition 2.1 (Regret). Let $a^* \in \arg \max_{a \in \mathcal{A}} \mathbf{E}_{y \sim P(\cdot|a)}[r(y)]$ denote the optimal arm. The T -period regret of the sequence of actions $a_1, \dots, a_T$ is the random variable

$$\mathbf{Regret}(T) = \sum_{t=1}^T \left[r(Y_t(a^*)) - r(Y_t(a_t)) \right].$$

Typically, MAB algorithms focus on the *expected regret*, $\mathbf{E}[\mathbf{Regret}(T)]$, and provide policies that are guaranteed to keep it small in some sense.

As an illustration of the MAB model, let us revisit the online shop example.

Example 2.1 (Online shop – bandit setting). Recall the online shop domain of Example 1.1. In the MAB setting, there is no state information

about the customers, i.e., $\mathcal{X} = \emptyset$. In addition, each interaction lasts a single time step, after which the customer checks out, and a profit is obtained according to his purchasing decision. The regret minimization problem corresponds to determining which sequence of advertisements $a_1, \dots, a_T$ to show to a stream of T incoming customers, such that the total revenue is close to that which would have been obtained with the optimal advertisement stream.

In some cases, the decision-maker may have access to some additional information that is important for making decisions, but is not captured in the MAB model. For example, in the online shop domain of Example 2.1, it is likely that some information about the customer, such as his age or origin, is available. The *contextual bandit* model (a.k.a. associative bandits, or bandits with side information) is an extension of the MAB model that takes such information into account.

The decision-making process in the contextual bandit model is similar to the MAB case. However, at each time step t , the decision maker first observes a context $s_t \in \mathcal{S}$, drawn i.i.d. over time from a probability distribution $P_S(\cdot) \in \mathcal{P}(\mathcal{S})$, where $\mathcal{P}(\mathcal{S})$ denotes the set of probability distributions on (Borel) subsets of $\mathcal{S}$. The decision-maker then chooses an action $a_t \in \mathcal{A}$ and observes a random outcome $Y_t(a_t, s_t) \in \mathcal{Y}$, which now depends both on the context and the action. The outcomes $\{Y_t(a)\}$ are drawn i.i.d. over time from an unknown probability distribution $P(\cdot|a, s) \in \mathcal{P}(\mathcal{Y})$, where $\mathcal{P}(\mathcal{Y})$ denotes the set of probability distributions on (Borel) subsets of $\mathcal{Y}$.

Model 2 (Contextual Bandit) Define a contextual bandit to be a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{Y}, P_S, P, r \rangle$ where

- $\mathcal{S}$ is the set of contexts,
- $\mathcal{A}$ is the set of actions (arms),
- $\mathcal{Y}$ is the set of possible outcomes,
- $P_S(\cdot) \in \mathcal{P}(\mathcal{S})$ is the context probability,
- $P(\cdot|a, s) \in \mathcal{P}(\mathcal{Y})$ is the outcome probability, conditioned on action $a \in \mathcal{A}$ being taken when the context is $s \in \mathcal{S}$,
- $r(Y) \in \mathbb{R}$ represents the reward obtained when outcome $Y \in \mathcal{Y}$ is observed.

The Markov decision process model, as presented in the following section, may be seen as an extension of the contextual bandit model to a *sequential* decision-making model, in which the context is no longer i.i.d., but may change over time according to the selected actions.

2.2 Markov Decision Processes

The Markov Decision Process (MDP) is a framework for sequential decision-making in Markovian dynamical systems [Bellman, 1957, Puterman, 1994]. It can be seen as an extension of the MAB framework by adding the notion of a system *state*, that may dynamically change according to the performed actions and affects the outcomes of the system.

Model 3 (Markov Decision Process) Define an MDP $\mathcal{M}$ to be a tuple $\langle \mathcal{S}, \mathcal{A}, P, P_0, q \rangle$ where

- $\mathcal{S}$ is the set of states,
- $\mathcal{A}$ is the set of actions,
- $P(\cdot|s, a) \in \mathcal{P}(\mathcal{S})$ is the probability distribution over next states, conditioned on action a being taken in state s ,
- $P_0 \in \mathcal{P}(\mathcal{S})$ is the probability distribution according to which the initial state is selected,
- $R(s, a) \sim q(\cdot|s, a) \in \mathcal{P}(\mathbb{R})$ is a random variable representing the reward obtained when action a is taken in state s .

Let $\mathcal{P}(\mathcal{S})$, $\mathcal{P}(\mathcal{A})$, and $\mathcal{P}(\mathbb{R})$ be the set of probability distributions on (Borel) subsets of $\mathcal{S}$, $\mathcal{A}$, and $\mathbb{R}$ respectively.¹ We assume that P , P_0 and q are stationary. Throughout the paper, we use upper-case and lower-case letters to refer to random variables and the values taken by random variables, respectively. For example, $R(s, a)$ is the random variable of the immediate reward, and $r(s, a)$ is one possible realization of this random variable. We denote the expected value of $R(s, a)$, as $\bar{r}(s, a) = \int r q(dr|s, a)$.

Assumption A1 (MDP Regularity) We assume that the random immediate rewards are bounded by $R_{\max}$ and the expected immediate

¹ $\mathbb{R}$ is the set of real numbers.

rewards are bounded by $\bar{R}_{\max}$. Note that $\bar{R}_{\max} \leq R_{\max}$.

A rule according to which the agent selects its actions at each possible state, or *policy*, is defined as a mapping from past observations to a distribution over the set of actions. A policy is called *Markov* if the distribution depends only on the last state of the observation sequence. A policy is called *stationary* if it does not change over time. A stationary Markov policy $\mu(\cdot|s) \in \mathcal{P}(\mathcal{A})$ is a probability distribution over the set of actions given a state $s \in \mathcal{S}$. A policy is *deterministic* if the probability distribution concentrates on a single action for all histories. A deterministic policy is identified by a mapping from the set of states to the set of actions, i.e., $\mu : \mathcal{S} \rightarrow \mathcal{A}$. In the rest of the paper, we use the term policy to refer to stationary Markov policies.

The MDP controlled by a policy μ induces a Markov chain $\mathcal{M}^\mu$ with reward distribution $q^\mu(\cdot|s) = q(\cdot|s, \mu(s))$ such that $R^\mu(s) = R(s, \mu(s)) \in q^\mu(\cdot|s)$, transition kernel $P^\mu(\cdot|s) = P(\cdot|s, \mu(s))$, and stationary distribution over states π^μ (if it admits one). In a Markov chain $\mathcal{M}^\mu$, for state-action pairs $z = (s, a) \in \mathcal{Z} = \mathcal{S} \times \mathcal{A}$, we define the transition density and the initial (state-action) density as $P^\mu(z'|z) = P(s'|s, a)\mu(a'|s')$ and $P_0^\mu(z_0) = P_0(s_0)\mu(a_0|s_0)$, respectively. We generically use $\xi = \{z_0, z_1, \dots, z_T\} \in \Xi$, $T \in \{0, 1, \dots, \infty\}$, to denote a path (or trajectory) generated by this Markov chain. The probability (density) of such a path is given by

$$\Pr(\xi|\mu) = P_0^\mu(z_0) \prod_{t=1}^T P^\mu(z_t|z_{t-1}).$$

We define the (possibly discounted, $\gamma \in [0, 1]$) return of a path as a function $\rho : \Xi \rightarrow \mathbb{R}$, $\rho(\xi) = \sum_{t=0}^T \gamma^t R(z_t)$. For each path ξ , its (discounted) return, $\rho(\xi)$, is a random variable with the expected value $\bar{\rho}(\xi) = \sum_{t=0}^T \gamma^t \bar{r}(z_t)$.² Here, $\gamma \in [0, 1]$ is a discount factor that determines the exponential devaluation rate of delayed rewards.³ The

²When there is no randomness in the rewards, i.e., $r(s, a) = \bar{r}(s, a)$, then $\rho(\xi) = \bar{\rho}(\xi)$.

³When $\gamma = 1$, the policy must be proper, i.e., guaranteed to terminate, see Puterman [1994] or Bertsekas and Tsitsiklis [1996] for more details.

expected return of a policy μ is defined by

$$\eta(\mu) = \mathbf{E}[\rho(\xi)] = \int_{\Xi} \bar{\rho}(\xi) \Pr(\xi|\mu) d\xi. \quad (2.1)$$

The expectation is over all possible trajectories generated by policy μ and all possible rewards collected in them.

Similarly, for a given policy μ , we can define the (discounted) return of a state s , $D^\mu(s)$, as the sum of (discounted) rewards that the agent encounters when it starts in state s and follows policy μ afterwards

$$D^\mu(s) = \sum_{t=0}^{\infty} \gamma^t R(Z_t) \mid Z_0 = (s, \mu(\cdot|s)), \quad \text{with } S_{t+1} \sim P^\mu(\cdot|S_t). \quad (2.2)$$

The expected value of D^μ is called the value function of policy μ

$$V^\mu(s) = \mathbf{E}[D^\mu(s)] = \mathbf{E} \left[\sum_{t=0}^{\infty} \gamma^t R(Z_t) \mid Z_0 = (s, \mu(\cdot|s)) \right]. \quad (2.3)$$

Closely related to value function is the action-value function of a policy, the total expected (discounted) reward observed by the agent when it starts in state s , takes action a , and then executes policy μ

$$Q^\mu(z) = \mathbf{E}[D^\mu(z)] = \mathbf{E} \left[\sum_{t=0}^{\infty} \gamma^t R(Z_t) \mid Z_0 = z \right],$$

where similarly to $D^\mu(s)$, $D^\mu(z)$ is the sum of (discounted) rewards that the agent encounters when it starts in state s , takes action a , and follows policy μ afterwards. It is easy to see that for any policy μ , the functions V^μ and Q^μ are bounded by $\bar{R}_{\max}/(1-\gamma)$. We may write the value of a state s under a policy μ in terms of its immediate reward and the values of its successor states under μ as

$$V^\mu(s) = R^\mu(s) + \gamma \int_{\mathcal{S}} P^\mu(s'|s) V^\mu(s') ds', \quad (2.4)$$

which is called the *Bellman equation* for V^μ .

Given an MDP, the goal is to find a policy that attains the best possible values, $V^*(s) = \sup_{\mu} V^\mu(s)$, for all states $s \in \mathcal{S}$. The function V^* is called the *optimal* value function. A policy is optimal (denoted by μ^*) if it attains the optimal values at all the states, i.e., $V^{\mu^*}(s) = V^*(s)$

for all $s \in \mathcal{S}$. In order to characterize optimal policies it is useful to define the optimal action-value function

$$Q^*(z) = \sup_{\mu} Q^{\mu}(z). \quad (2.5)$$

Further, we say that a deterministic policy μ is *greedy* with respect to an action-value function Q , if for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$, $\mu(s) \in \arg \max_{a \in \mathcal{A}} Q(s, a)$. Greedy policies are important because any greedy policy with respect to Q^* is optimal. Similar to the value function of a policy (Eq. 2.4), the optimal value function of a state s may be written in terms of the optimal values of its successor states as

$$V^*(s) = \max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} P(s'|s, a) V^*(s') ds' \right], \quad (2.6)$$

which is called the *Bellman optimality equation*. Note that, similar to Eqs. 2.4 and 2.6, we may define the Bellman equation and the Bellman optimality equation for action-value function.

It is important to note that almost all methods for finding the optimal solution of an MDP are based on two *dynamic programming* (DP) algorithms: *value iteration* and *policy iteration*. Value iteration (VI) begins with a value function V_0 and at each iteration i , generates a new value function by applying Eq. 2.6 to the current value function, i.e.,

$$V_i^*(s) = \max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \int_{\mathcal{S}} P(s'|s, a) V_{i-1}^*(s') ds' \right], \quad \forall s \in \mathcal{S}.$$

Policy iteration (PI) starts with an initial policy. At each iteration, it evaluates the value function of the current policy, this process is referred to as *policy evaluation* (PE), and then performs a *policy improvement* step, in which a new policy is generated as a greedy policy with respect to the value of the current policy. Iterating the policy evaluation – policy improvement process is known to produce a strictly monotonically improving sequence of policies. If the improved policy is the same as the policy improved upon, then we are assured that the optimal policy has been found.

2.3 Partially Observable Markov Decision Processes

The Partially Observable Markov Decision Process (POMDP) is an extension of MDP to the case where the state of the system is not necessarily observable [Astrom, 1965, Smallwood and Sondik, 1973, Kaelbling et al., 1998].

Model 4 (Partially Observable Markov Decision Process) Define a POMDP $\mathcal{M}$ to be a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, P, \Omega, P_0, q \rangle$ where

- $\mathcal{S}$ is the set of states,
- $\mathcal{A}$ is the set of actions,
- $\mathcal{O}$ is the set of observations,
- $P(\cdot|s, a) \in \mathcal{P}(\mathcal{S})$ is the probability distribution over next states, conditioned on action a being taken in state s ,
- $\Omega(\cdot|s, a) \in \mathcal{P}(\mathcal{O})$ is the probability distribution over possible observations, conditioned on action a being taken to reach state s where the observation is perceived,
- $P_0 \in \mathcal{P}(\mathcal{S})$ is the probability distribution according to which the initial state is selected,
- $R(s, a) \sim q(\cdot|s, a) \in \mathcal{P}(\mathbb{R})$ is a random variable representing the reward obtained when action a is taken in state s .

All assumptions are similar to MDPs, with the addition that $\mathcal{P}(\mathcal{O})$ is the set of probability distributions on (Borel) subsets of $\mathcal{O}$, and Ω is stationary. As a motivation for the POMDP model, we revisit again the online shop domain.

Example 2.2 (Online shop – hidden state setting). Recall the online shop domain of Example 1.1. In a realistic scenario, some features of the customer, such as its gender or age, might not be visible to the decision maker due to privacy, or other reasons. In such a case, the MDP model, which requires the full state information for making decisions is not suitable. In the POMDP model, only observable quantities, such as the items in the shopping cart, are used for making decisions.

Since the state is not directly observed, the agent must rely on the recent history of actions and observations, $\{o_{t+1}, a_t, o_t, \dots, o_1, a_0\}$

to infer a distribution over states. This *belief* (also called *information state*) is defined over the state probability simplex, $b_t \in \Delta$, and can be calculated recursively as:

$$b_{t+1}(s') = \frac{\Omega(o_{t+1}|s', a_t) \int_{\mathcal{S}} P(s'|s, a_t) b_t(s) ds}{\int_{\mathcal{S}} \Omega(o_{t+1}|s'', a_t) \int_{\mathcal{S}} P(s''|s, a_t) b_t(s) ds ds''} , \quad (2.7)$$

where the sequence is initialized at $b_0 := P_0$ and the denominator can be seen as a simple normalization function. For convenience, we sometimes denote the belief update (Eq. 2.7) as $b_{t+1} = \tau(b_t, a, o)$.

In the POMDP framework, the action-selection policy is defined as $\mu : \Delta(\mathcal{S}) \rightarrow \mathcal{A}$. Thus, solving a POMDP involves finding the optimal policy, μ^* , that maximizes the expected discounted sum of rewards for all belief states. This can be defined using a variant of the Bellman equation

$$V^*(b_t) = \max_{a \in \mathcal{A}} \left[\int_{\mathcal{S}} R(s, a) b_t(s) ds + \gamma \int_{\mathcal{O}} \Pr(o|b_t, a) V^*(\tau(b_t, a, o)) do \right] . \quad (2.8)$$

The optimal value function for a finite-horizon POMDP can be shown to be piecewise-linear and convex [Smallwood and Sondik, 1973, Porta et al., 2006]. In that case, the value function V_t at any finite planning horizon t can be represented by a finite set of linear segments $\Gamma_t = \{\alpha_0, \alpha_1, \dots, \alpha_m\}$, often called α -vectors (when $\mathcal{S}$ is discrete) or α -functions (when $\mathcal{S}$ is continuous). Each defines a linear function over the belief state space associated with some action $a \in \mathcal{A}$. The value of a given α_i at a belief b_t can be evaluated by linear interpolation

$$\alpha_i(b_t) = \int_{\mathcal{S}} \alpha_i(s) b_t(s) ds. \quad (2.9)$$

The value of a belief state is the maximum value returned by one of the α -functions for this belief state

$$V_t^*(b_t) = \max_{\alpha \in \Gamma_t} \int_{\mathcal{S}} \alpha(s) b_t(s) . \quad (2.10)$$

In that case, the best action, $\mu^*(b_t)$, is the one associated with the α -vector that returns the best value.

The belief as defined here can be interpreted as a state in a particular kind of MDP, often called the *belief-MDP*. The main intuition is that

for any partially observable system with known parameters, the belief is actually fully observable and computable (for small enough state spaces and is approximable for others). Thus, the planning problem is in fact one of planning in an MDP, where the state space corresponds to the belief simplex. This does not lead to any computational advantage, but conceptually, suggests that known results and methods for MDPs can also be applied to the belief-MDP.

2.4 Reinforcement Learning

Reinforcement learning (RL) [Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998] is a class of learning *problems* in which an agent (or controller) interacts with a dynamic, stochastic, and incompletely known environment, with the goal of finding an action-selection strategy, or *policy*, to optimize some measure of its long-term performance. The interaction is conventionally modeled as an MDP, or if the environment state is not always completely observable, as a POMDP [Puterman, 1994].

Now that we have defined the MDP model, the next step is to solve it, i.e., to find an optimal policy.⁴ In some cases, MDPs can be solved analytically, and in many cases they can be solved iteratively by *dynamic* or *linear programming* (e.g., see Bertsekas and Tsitsiklis [1996]). However, in other cases these methods cannot be applied because either the state space is too large, a system model is available only as a simulator, or no system model is available at all. It is in these cases that RL techniques and algorithms may be helpful.

Reinforcement learning solutions can be viewed as a broad class of sample-based methods for solving MDPs. In place of a model, these methods use sample trajectories of the system and the agent interacting, such as could be obtained from a simulation. It is not unusual in practical applications for such a simulator to be available when an explicit transition-probability model of the sort suitable for use by dynamic or linear programming is not [Tesauro, 1994, Crites and Barto,

⁴It is not possible to find an optimal policy in many practical problems. In such cases, the goal would be to find a reasonably good policy, i.e., a policy with large enough value function.

1998]. Reinforcement learning methods can also be used with no model at all, by obtaining sample trajectories from direct interaction with the system [Baxter et al., 1998, Kohl and Stone, 2004, Ng et al., 2004].

Reinforcement learning solutions can be categorized in different ways. Below we describe two common categorizations that are relevant to the structure of this paper.

Model-based vs. Model-free Methods: Reinforcement learning algorithms that explicitly learn a system model and use it to solve an MDP are called *model-based* methods. Some examples of these methods are the Dyna architecture [Sutton, 1991] and prioritized sweeping [Moore and Atkeson, 1993]. *Model-free* methods are those that do not explicitly learn a system model and only use sample trajectories obtained by direct interaction with the system. These methods include popular algorithms such as Q-learning [Watkins, 1989], SARSA [Rummery and Niranjan, 1994], and LSPI [Lagoudakis and Parr, 2003].

Value Function vs. Policy Search Methods: An important class of RL algorithms are those that first find the optimal value function, and then extract an optimal policy from it. This class of RL algorithms contains *value function* methods, of which some well-studied examples include value iteration [Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998]), policy iteration [Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998]), Q-learning [Watkins, 1989], SARSA [Rummery and Niranjan, 1994], LSPI [Lagoudakis and Parr, 2003], and fitted Q-iteration [Ernst et al., 2005]. An alternative approach for solving an MDP is to directly search in the space of policies. These RL algorithms are called *policy search* methods. Since the number of policies is exponential in the size of the state space, one has to resort to meta-heuristic search [Mannor et al., 2003] or to local greedy methods. An important class of policy search methods is that of *policy gradient* algorithms. In these algorithms, the policy is taken to be an arbitrary differentiable function of a parameter vector, and the search in the policy space is directed by the gradient of a performance function with

respect to the policy parameters [Williams, 1992, Marbach, 1998, Baxter and Bartlett, 2001]). There is a third class of RL methods that use policy gradient to search in the policy space, and at the same time estimate a value function. These algorithms are called *actor-critic* [Konda and Tsitsiklis, 2000, Sutton et al., 2000, Bhatnagar et al., 2007, Peters and Schaal, 2008, Bhatnagar et al., 2009]). They can be thought of as RL analogs of dynamic programming’s policy iteration method. Actor-critic methods are based on the simultaneous online estimation of the parameters of two structures, called the *actor* and the *critic*. The actor and the critic correspond to conventional action-selection policy and value function, respectively. These problems are separable, but are solved simultaneously to find an optimal policy.

2.5 Bayesian Learning

In Bayesian learning, we make inference about a random variable X by producing a probability distribution for X . Inferences, such as point and interval estimates, may then be extracted from this distribution. Let us assume that the random variable X is hidden and we can only observe a related random variable Y . Our goal is to infer X from the samples of Y . A simple example is when X is a physical quantity and Y is its noisy measurement. Bayesian inference is usually carried out in the following way:

1. We choose a probability density $P(X)$, called the *prior distribution*, that expresses our beliefs about the random variable X before we observe any data.
2. We select a statistical model $P(Y|X)$ that reflects our belief about Y given X . This model represents the statistical dependence between X and Y .
3. We observe data $Y = y$.
4. We update our belief about X by calculating its posterior distribution using Bayes rule

$$P(X|Y = y) = \frac{P(y|X)P(X)}{\int P(y|X')P(X')dX'}.$$

Assume now that $P(X)$ is parameterized by an unknown vector of parameters θ in some parameter space Θ ; we denote this as $P_\theta(X)$. Let $X_1, \dots, X_n$ be a random i.i.d. sample drawn from $P_\theta(X)$. In general, updating the posterior $P_\theta(X|Y = y)$ is difficult, due to the need to compute the normalizing constant $\int_\Theta P(y|X')P_\theta(X')d\theta$. However, for the case of *conjugate* family distributions, we can update the posterior in closed-form by simply updating the parameters of the distribution. In the next two sections, we consider three classes of conjugate distributions: Beta and Dirichlet distributions and Gaussian Processes (GPs).

2.5.1 Beta and Dirichlet Distributions

A simple example of a conjugate family is the Beta distribution, which is conjugate to the Binomial distribution. Let $Beta(\alpha, \beta)$ be defined by the density function $f(p|\alpha, \beta) \propto p^{\alpha-1}(1-p)^{\beta-1}$ for $p \in [0, 1]$, and parameters $\alpha, \beta \geq 0$. Let $Binomial(n, p)$ be defined by the density function $f(k|n, p) \propto p^k(1-p)^{n-k}$ for $k \in \{0, 1, \dots, n\}$, and parameters $p \in [0, 1]$ and $n \in \mathbb{N}$. Consider $X \sim Binomial(n, p)$ with unknown probability parameter p , and consider a prior $Beta(\alpha, \beta)$ over the unknown value of p . Then following an observation $X = x$, the posterior over p is also Beta distributed and is defined by $Beta(\alpha + x, \beta + n - x)$.

Now let us consider the multivariate extension of this conjugate family. In this case, we have the Multinomial distribution, whose conjugate is the Dirichlet distribution. Let $X \sim Multinomial_k(p, N)$ be a random variable with unknown probability distribution $p = (p_1, \dots, p_k)$. The Dirichlet distribution is parameterized by a count vector $\phi = (\phi_1, \dots, \phi_k)$, where $\phi_i \geq 0$, such that the density of probability distribution $p = (p_1, \dots, p_k)$ is defined as $f(p|\phi) \propto \prod_{i=1}^k p_i^{\phi_i-1}$. The distribution $Dirichlet(\phi_1, \dots, \phi_k)$ can also be interpreted as a prior over the unknown Multinomial distribution p , such that after observing $X = \mathbf{n}$, the posterior over p is also Dirichlet and is defined by $Dirichlet(\phi_1 + n_1, \dots, \phi_k + n_k)$.

2.5.2 Gaussian Processes and Gaussian Process Regression

A Gaussian process (GP) is an indexed set of jointly Gaussian random variables, i.e., $F(x)$, $x \in \mathcal{X}$ is a Gaussian process if and only

if for every finite set of indices $\{x_1, \dots, x_T\}$ in the index set $\mathcal{X}$, $(F(x_1), \dots, F(x_T))$ is a vector-valued Gaussian random variable. The GP F may be thought of as a random vector if $\mathcal{X}$ is finite, as a random series if $\mathcal{X}$ is countably infinite, and as a random function if $\mathcal{X}$ is uncountably infinite. In the last case, each instantiation of F is a function $f : \mathcal{X} \rightarrow \mathbb{R}$. For a given x , $F(x)$ is a random variable, normally distributed jointly with the other components of F . A GP F can be fully specified by its mean $\bar{f} : \mathcal{X} \rightarrow \mathbb{R}$, $\bar{f}(x) = \mathbf{E}[F(x)]$ and covariance $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, $k(x, x') = \mathbf{Cov}[F(x), F(x')]$, i.e., $F(\cdot) \sim \mathcal{N}(\bar{f}(\cdot), k(\cdot, \cdot))$. The kernel function $k(\cdot, \cdot)$ encodes our prior knowledge concerning the correlations between the components of F at different points. It may be thought of as inducing a measure of proximity between the members of $\mathcal{X}$. It can also be shown that k defines the function space within which the search for a solution takes place (see Schölkopf and Smola [2002] and Shawe-Taylor and Cristianini [2004] for more details).

Now let us consider the following generative model:

$$Y(x) = \mathbf{H}F(x) + N(x), \quad (2.11)$$

where $\mathbf{H}$ is a general linear transformation, F is an unknown function, x is the input, N is a Gaussian noise and independent of F , and Y is the observable process, modeled as a noisy version of $\mathbf{H}F$. The objective here is to infer F from samples of Y . Bayesian learning can be applied to this problem in the following way.

1. We choose a prior distribution for F in the form of a GP, $F(\cdot) \sim \mathcal{N}(\bar{f}(\cdot), k(\cdot, \cdot))$. When F and N are Gaussian and independent of each other, the generative model of Eq. 2.11 is known as the *linear statistical model* [Scharf, 1991].
2. The statistical dependence between F and Y is defined by the model-equation (2.11).
3. We observe a sample in the form of $\mathcal{D}_T = \{(x_t, y_t)\}_{t=1}^T$.
4. We calculate the posterior distribution of F conditioned on the sample $\mathcal{D}_T$ using the Bayes rule. Below is the process to calculate this posterior distribution.

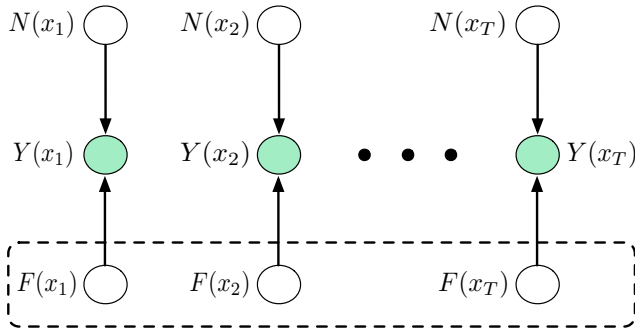


Figure 2.1: A directed graph illustrating the conditional independencies between the latent $F(x_t)$ variables (bottom row), the noise variables $N(x_t)$ (top row), and the observable $Y(x_t)$ variables (middle row), in GP regression (when $\mathbf{H} = \mathbf{I}$). All of the $F(x_t)$ variables should be interconnected by arrows (forming a clique), due to the dependencies introduced by the prior. To avoid cluttering the diagram, this was marked by the dashed frame surrounding them.

Figure 2.1 illustrates the GP regression setting (when $\mathbf{H} = \mathbf{I}$) as a graphical model in which arrows mark the conditional dependency relations between the nodes corresponding to the latent $F(x_t)$ and the observed $Y(x_t)$ variables. The model-equation (2.11) evaluated at the training samples may be written as

$$\mathbf{Y}_T = \mathbf{H}\mathbf{F}_T + \mathbf{N}_T, \quad (2.12)$$

where $\mathbf{F}_T = (F(x_1), \dots, F(x_T))^\top$, $\mathbf{Y}_T = (y_1, \dots, y_T)^\top$, and $\mathbf{N}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. Here $[\mathbf{\Sigma}]_{i,j}$ is the measurement noise covariance between the i th and j th samples. In the linear statistical model, we then have $\mathbf{F}_T \sim \mathcal{N}(\bar{\mathbf{f}}, \mathbf{K})$, where $\bar{\mathbf{f}} = (\bar{f}(x_1), \dots, \bar{f}(x_T))^\top$ and $[\mathbf{K}]_{i,j} = k(x_i, x_j)$ is a $T \times T$ kernel matrix. Since both $\mathbf{F}_T$ and $\mathbf{N}_T$ are Gaussian and independent from each other, we have $\mathbf{Y}_T \sim \mathcal{N}(\mathbf{H}\bar{\mathbf{f}}, \mathbf{H}\mathbf{K}\mathbf{H}^\top + \mathbf{\Sigma})$. Consider a query point x , we then have

$$\begin{pmatrix} F(x) \\ \mathbf{F}_T \\ \mathbf{N}_T \end{pmatrix} = \mathcal{N} \left\{ \begin{pmatrix} \bar{f}(x) \\ \bar{\mathbf{f}} \\ \mathbf{0} \end{pmatrix}, \begin{bmatrix} k(x, x) & \mathbf{k}(x)^\top & \mathbf{0} \\ \mathbf{k}(x) & \mathbf{K} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Sigma} \end{bmatrix} \right\},$$

where $\mathbf{k}(x) = (k(x_1, x), \dots, k(x_T, x))^\top$. Using Eq. 2.12, we have the

following transformation:

$$\begin{pmatrix} F(x) \\ \mathbf{F}_T \\ \mathbf{Y}_T \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \mathbf{I} & \mathbf{0} \\ 0 & \mathbf{H} & \mathbf{I} \end{bmatrix} \begin{pmatrix} F(x) \\ \mathbf{F}_T \\ \mathbf{N}_T \end{pmatrix}, \quad (2.13)$$

where $\mathbf{I}$ is the identity matrix. From Eq. 2.13, we have

$$\begin{pmatrix} F(x) \\ \mathbf{F}_T \\ \mathbf{Y}_T \end{pmatrix} = \mathcal{N} \left\{ \begin{pmatrix} \bar{f}(x) \\ \bar{\mathbf{f}} \\ \mathbf{H}\bar{\mathbf{f}} \end{pmatrix}, \begin{bmatrix} k(x, x) & \mathbf{k}(x)^\top & \mathbf{k}(x)^\top \mathbf{H}^\top \\ \mathbf{k}(x) & \mathbf{K} & \mathbf{K} \mathbf{H}^\top \\ \mathbf{H} \mathbf{k}(x) & \mathbf{H} \mathbf{K} & \mathbf{H} \mathbf{K} \mathbf{H}^\top + \boldsymbol{\Sigma} \end{bmatrix} \right\}.$$

Using the Gauss-Markov theorem [Scharf, 1991], we know that $F(x)|\mathbf{Y}_T$ (or equivalently $F(x)|\mathcal{D}_T$) is Gaussian, and obtain the following expressions for the posterior mean and covariance of $F(x)$ conditioned on the sample $\mathcal{D}_T$:

$$\begin{aligned} \mathbf{E}[F(x)|\mathcal{D}_T] &= \bar{f}(x) + \mathbf{k}(x)^\top \mathbf{H}^\top (\mathbf{H} \mathbf{K} \mathbf{H}^\top + \boldsymbol{\Sigma})^{-1} (\mathbf{y}_T - \mathbf{H} \bar{\mathbf{f}}), \\ \mathbf{Cov}[F(x), F(x')|\mathcal{D}_T] &= k(x, x') - \mathbf{k}(x)^\top \mathbf{H}^\top (\mathbf{H} \mathbf{K} \mathbf{H}^\top + \boldsymbol{\Sigma})^{-1} \mathbf{H} \mathbf{k}(x'), \end{aligned} \quad (2.14)$$

where $\mathbf{y}_T = (y_1, \dots, y_T)^\top$ is one realization of the random vector $\mathbf{Y}_T$. It is possible to decompose the expressions in Eq. 2.14 into input dependent terms (which depend on x and x') and terms that only depend on the training samples, as follows:

$$\begin{aligned} \mathbf{E}[F(x)|\mathcal{D}_T] &= \bar{f}(x) + \mathbf{k}(x)^\top \boldsymbol{\alpha}, \\ \mathbf{Cov}[F(x), F(x')|\mathcal{D}_T] &= k(x, x') - \mathbf{k}(x)^\top \mathbf{C} \mathbf{k}(x'), \end{aligned} \quad (2.15)$$

where

$$\begin{aligned} \boldsymbol{\alpha} &= \mathbf{H}^\top (\mathbf{H} \mathbf{K} \mathbf{H}^\top + \boldsymbol{\Sigma})^{-1} (\mathbf{y}_T - \mathbf{H} \bar{\mathbf{f}}), \\ \mathbf{C} &= \mathbf{H}^\top (\mathbf{H} \mathbf{K} \mathbf{H}^\top + \boldsymbol{\Sigma})^{-1} \mathbf{H}. \end{aligned} \quad (2.16)$$

From Eqs. 2.15 and 2.16, we can conclude that $\boldsymbol{\alpha}$ and $\mathbf{C}$ are *sufficient statistics* for the posterior moments.

If we set the transformation $\mathbf{H}$ to be the identity and assume that the noise terms corrupting each sample are i.i.d. Gaussian, i.e., $\mathbf{N}_T \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$, where σ^2 is the variance of each noise term, the linear statistical model is reduced to the *standard linear regression model*. In this

case, the posterior moments of $F(x)$ can be written as

$$\begin{aligned}\mathbf{E}[F(x)|\mathcal{D}_T] &= \bar{f}(x) + \mathbf{k}(x)^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} (\mathbf{y}_T - \bar{\mathbf{f}}) \\ &= \bar{f}(x) + \mathbf{k}(x)^\top \boldsymbol{\alpha}, \\ \mathbf{Cov}[F(x), F(x')|\mathcal{D}_T] &= k(x, x') - \mathbf{k}(x)^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}(x') \\ &= k(x, x') - \mathbf{k}(x)^\top \mathbf{C} \mathbf{k}(x'),\end{aligned}\tag{2.17}$$

with

$$\boldsymbol{\alpha} = (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} (\mathbf{y}_T - \bar{\mathbf{f}}), \quad \mathbf{C} = (\mathbf{K} + \sigma^2 \mathbf{I})^{-1}.\tag{2.18}$$

The GP regression described above is kernel-based and non-parametric. It is also possible to employ a parametric representation under very similar assumptions. In the parametric setting, the GP F is assumed to consist of a linear combination of a finite number n of basis functions $\varphi_i : \mathcal{X} \rightarrow \mathbb{R}$, $i = 1, \dots, n$. In this case, F can be written as $F(\cdot) = \sum_{i=1}^n \varphi_i(\cdot) W_i = \boldsymbol{\phi}(\cdot)^\top \mathbf{W}$, where $\boldsymbol{\phi}(\cdot) = (\varphi_1(\cdot), \dots, \varphi_n(\cdot))^\top$ is the feature vector and $\mathbf{W} = (W_1, \dots, W_n)^\top$ is the weight vector. The model-equation (2.11) now becomes

$$Y(x) = \mathbf{H} \boldsymbol{\phi}(x)^\top \mathbf{W} + N(x).$$

In the parametric GP regression, the randomness in F is due to $\mathbf{W}$ being a random vector. Here we consider a Gaussian prior over $\mathbf{W}$, distributed as $\mathbf{W} \sim \mathcal{N}(\bar{\mathbf{w}}, \mathbf{S}_w)$. By applying the Bayes rule, the posterior (Gaussian) distribution of $\mathbf{W}$ conditioned on the observed data $\mathcal{D}_T$ can be computed as

$$\begin{aligned}\mathbf{E}[\mathbf{W}|\mathcal{D}_T] &= \bar{\mathbf{w}} + \mathbf{S}_w \boldsymbol{\Phi} \mathbf{H}^\top (\mathbf{H} \boldsymbol{\Phi}^\top \mathbf{S}_w \boldsymbol{\Phi} \mathbf{H}^\top + \boldsymbol{\Sigma})^{-1} (\mathbf{y}_T - \mathbf{H} \boldsymbol{\Phi}^\top \bar{\mathbf{w}}), \\ \mathbf{Cov}[\mathbf{W}|\mathcal{D}_T] &= \mathbf{S}_w - \mathbf{S}_w \boldsymbol{\Phi} \mathbf{H}^\top (\mathbf{H} \boldsymbol{\Phi}^\top \mathbf{S}_w \boldsymbol{\Phi} \mathbf{H}^\top + \boldsymbol{\Sigma})^{-1} \mathbf{H} \boldsymbol{\Phi}^\top \mathbf{S}_w,\end{aligned}\tag{2.19}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\phi}(x_1), \dots, \boldsymbol{\phi}(x_T)]$ is a $n \times T$ feature matrix. Finally, since $F(x) = \boldsymbol{\phi}(x)^\top \mathbf{W}$, the posterior mean and covariance of F can be easily

computed as

$$\begin{aligned}
\mathbf{E}[F(x)|\mathcal{D}_T] &= \phi(x)^\top \bar{\mathbf{w}} \\
&\quad + \phi(x)^\top \mathbf{S}_{\mathbf{w}} \Phi \mathbf{H}^\top (\mathbf{H} \Phi^\top \mathbf{S}_{\mathbf{w}} \Phi \mathbf{H}^\top + \Sigma)^{-1} (\mathbf{y}_T - \mathbf{H} \Phi^\top \bar{\mathbf{w}}), \\
\mathbf{Cov}[F(x), F(x')|\mathcal{D}_T] &= \phi(x)^\top \mathbf{S}_{\mathbf{w}} \phi(x') \\
&\quad - \phi(x)^\top \mathbf{S}_{\mathbf{w}} \Phi \mathbf{H}^\top (\mathbf{H} \Phi^\top \mathbf{S}_{\mathbf{w}} \Phi \mathbf{H}^\top + \Sigma)^{-1} \mathbf{H} \Phi^\top \mathbf{S}_{\mathbf{w}} \phi(x').
\end{aligned} \tag{2.20}$$

Similar to the non-parametric setting, for the standard linear regression model with the prior $\mathbf{W} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, Eq. 2.19 may be written as

$$\begin{aligned}
\mathbf{E}[\mathbf{W}|\mathcal{D}_T] &= \Phi(\Phi^\top \Phi + \sigma^2 \mathbf{I})^{-1} \mathbf{y}_T, \\
\mathbf{Cov}[\mathbf{W}|\mathcal{D}_T] &= \mathbf{I} - \Phi(\Phi^\top \Phi + \sigma^2 \mathbf{I})^{-1} \Phi^\top.
\end{aligned} \tag{2.21}$$

To have a smaller matrix inversion when $T > n$, Eq. 2.21 may be written as

$$\begin{aligned}
\mathbf{E}[\mathbf{W}|\mathcal{D}_T] &= (\Phi \Phi^\top + \sigma^2 \mathbf{I})^{-1} \Phi \mathbf{y}_T, \\
\mathbf{Cov}[\mathbf{W}|\mathcal{D}_T] &= \sigma^2 (\Phi \Phi^\top + \sigma^2 \mathbf{I})^{-1}.
\end{aligned} \tag{2.22}$$

For more details on GPs and GP regression, see Rasmussen and Williams [2006].

3

Bayesian Bandits

This section focuses on Bayesian learning methods for regret minimization in the multi-armed bandits (MAB) model. We review classic performance bounds for this problem and state-of-the-art results for several Bayesian approaches.

In the MAB model (§2.1), the only unknown quantities are the outcome probabilities $P(\cdot|a)$. The idea behind Bayesian approaches to MAB is to use Bayesian inference (§2.5) for learning $P(\cdot|a)$ from the outcomes observed during sequential interaction with the MAB. Following the framework of §2.5, the outcome probabilities are parameterized by an unknown vector θ , and are henceforth denoted by $P_\theta(\cdot|a)$. The parameter vector θ is assumed to be drawn from a prior distribution P_{prior} . As a concrete example, consider a MAB with Bernoulli arms:

Example 3.1 (Bernoulli K -MAB with a Beta prior). Consider a K -MAB where the set of outcomes is binary $\mathcal{Y} = \{0, 1\}$, and the reward is the identity $r(Y) = Y$. Let the outcome probabilities be parameterized by $\theta \in \mathbb{R}^K$, such that $Y(a) \sim \text{Bernoulli}[\theta_a]$. The prior for each θ_a is $\text{Beta}(\alpha_a, \beta_a)$. Following an observation outcome $Y(a) = y$, the posterior for θ_a is updated to $\text{Beta}(\alpha_a + y, \beta_a + y - 1)$. Also note that the (posterior) expected reward for arm a is now $\mathbf{E}[r(Y(a))]$ = $\mathbf{E}[\mathbf{E}[Y(a)|\theta_a]]$ = $\mathbf{E}[\theta_a]$ = $\frac{\alpha_a + y}{\alpha_a + y + \beta_a + y - 1}$.

The principal challenge of Bayesian MAB algorithms, however, is to utilize the posterior θ for selecting an adequate policy that achieves low regret.

Conceptually, there are several reasons to adopt a Bayesian approach for MABs. First, from a modelling perspective, the prior for θ is a natural means for embedding prior knowledge, or structure of the problem. Second, the posterior for θ encodes the uncertainty about the outcome probabilities at each step of the algorithm, and may be used for guiding the exploration to more relevant areas.

A Bayesian point of view may also be taken towards the performance measure of the algorithm. Recall that the performance of a MAB algorithm is measured by its expected regret. This expectation, however, may be defined in two different ways, depending on how the parameters θ are treated. In the frequentist approach, henceforth termed ‘*frequentist regret*’ and denoted $\mathbf{E}_{\theta}[\text{Regret}(T)]$, the parameter vector θ is fixed, and treated as a constant in the expectation. The expectation is then computed with respect to the stochastic outcomes and the action selection policy. On the other hand, in ‘*Bayesian regret*’, a.k.a. *Bayes risk*, θ is assumed to be drawn from the prior, and the expectation $\mathbf{E}[\text{Regret}(T)]$ is over the stochastic outcomes, the action selection rule, and the prior distribution of θ . Note that the optimal action a^* depends on θ . Therefore, in the expectation it is also considered a random variable.

We emphasize the separation of Bayesian MAB algorithms and Bayesian regret analysis. In particular, the performance of a Bayesian MAB algorithm (i.e., an algorithm that uses Bayesian learning techniques) may be measured with respect to a frequentist regret, or a Bayesian one.

3.1 Classical Results

In their seminal work, Lai and Robbins [1985] proved asymptotically tight bounds on the frequentist regret in terms of the Kullback-Leibler (KL) divergence between the distributions of the rewards of the different arms. These bounds grow logarithmically with the number of

steps T , such that regret is $\mathcal{O}(\ln T)$. Mannor and Tsitsiklis [2004] later showed non-asymptotic lower bounds with a similar logarithmic dependence on T . For the Bayesian regret, the lower bound on the regret is $\mathcal{O}(\sqrt{KT})$ (see, e.g., Theorem 3.5 of Bubeck and Cesa-Bianchi [2012]).

In the Bayesian setting, and for models that admit sufficient statistics, Gittins [1979] showed that an optimal strategy may be found by solving a specific MDP planning problem. The key observation here is that the dynamics of the posterior for each arm may be represented by a special MDP termed a *bandit process* [Gittins, 1979].

Definition 3.1 (Bandit Process). A bandit process is an MDP with two actions $\mathcal{A} = \{0, 1\}$. The control 0 freezes the process in the sense that $P(s' = s | s, 0) = 1$, and $R(s, 0) = 0$. Control 1 continues the process, and induces a standard MDP transition to a new state with probability $P(\cdot | s, 1)$, and reward $R(s, 1)$.

For example, consider the case of Bernoulli bandits with a Beta prior as described in Example 3.1. We identify the state s_a of the bandit process for arm a as the posterior parameters $s_a = (\alpha_a, \beta_a)$. Whenever arm a is pulled, the continuation control is applied as follows: we draw some θ_a from the posterior, and then draw an outcome $Y \sim \text{Bernoulli}[\theta_a]$. The state subsequently transitions to an updated posterior (cf. Example 3.1) $s'_a = (\alpha_a + Y, \beta_a + Y - 1)$, and a reward of $\mathbf{E}[r(Y)]$ is obtained, where the expectation is taken over the posterior.

The K -MAB problem, thus, may be seen as a particular instance of a general model termed *simple family of alternative bandit processes* (SFABP) [Gittins, 1979].

Model 5 (Simple Family of Alternative Bandit Processes)

A simple family of alternative bandit processes is a set of K bandit processes. For each bandit process $i \in 1, \dots, K$ we denote by s_i , $P_i(\cdot | s_i, 1)$, and $R_i(s_i, 1)$ the corresponding state, transition probabilities, and reward, respectively. At each time $t = 1, 2, \dots$, a single bandit $i_t \in 1, \dots, K$ that is in state $s_{i_t}(t)$ is activated, by applying to it the continuation control, and all other bandits are frozen. The objective is to find a bandit selection policy that maximizes the expected total

γ -discounted reward

$$\mathbf{E} \left[\sum_{t=0}^{\infty} \gamma^t R_{i_t}(s_{i_t}(t), 1) \right].$$

An important observation is that for the K -MAB problem, as defined above, the SFABP performance measure captures an expectation of the total discounted reward with respect to the *prior* distribution of the parameters θ . In this sense, this approach is a *full* Bayesian K -MAB solution (cf. the Bayesian regret vs. frequentist regret discussion above). In particular, note that the SFABP performance measure implicitly balances exploration and exploitation. To see this, recall the Bernoulli bandits example, and note that the reward of each bandit process is the *posterior* expected reward of its corresponding arm. Since the posterior distribution is encoded in the process state, future rewards in the SFABP performance measure depend on the future state of knowledge, thereby implicitly quantifying the value of additional observations for each arm.

The SFABP may be seen as a single MDP, with a state $(s_1, s_2, \dots, s_K)$ that is the conjunction of the K individual bandit process states. Thus, naively, an optimal policy may be computed by solving this MDP. Unfortunately, since the size of the state space of this MDP is exponential in K , such a naive approach is intractable. The virtue of the celebrated Gittins index theorem [Gittins, 1979], is to show that this problem nevertheless has a tractable solution.

Theorem 3.1. [Gittins, 1979] The objective of SFABP is maximized by following a policy that always chooses the bandit with the largest Gittins index

$$G_i(s_i) = \sup_{\tau \geq 1} \frac{\mathbf{E} \left[\sum_{t=0}^{\tau-1} \gamma^t R_i(s_i(t), 1) \mid s_i(0) = s_i \right]}{\mathbf{E} \left[\sum_{t=0}^{\tau-1} \gamma^t \mid s_i(0) = s_i \right]},$$

where τ is any (past measurable) stopping time.

The crucial advantage of Theorem 3.1, is that calculating G_i may be done *separately* for each arm, thereby avoiding the exponential complexity in K . The explicit calculation, however, is technically involved,

and beyond the scope of this survey. For reference see Gittins [1979], and also Tsitsiklis [1994] for a simpler derivation, and Niño-Mora [2011] for a finite horizon setting.

Due to the technical complexity of calculating optimal Gittin’s index policies, recent approaches concentrate on much simpler algorithms, that nonetheless admit optimal upper bounds (i.e., match the order of their respective lower bounds up to constant factors) on the expected regret.

3.2 Bayes-UCB

The upper confidence bound (UCB) algorithm [Auer et al., 2002] is a popular frequentist approach for MABs that employs an ‘optimistic’ policy to reduce the chance of overlooking the best arm. The algorithm starts by playing each arm once. Then, at each time step t , UCB plays the arm a that maximizes $\langle r_a \rangle + \sqrt{\frac{2 \ln t}{t_a}}$, where $\langle r_a \rangle$ is the average reward obtained from arm a , and t_a is the number of times arm a has been playing so far. The optimistic upper confidence term $\sqrt{\frac{2 \ln t}{t_a}}$ guarantees that the empirical average does not underestimate the best arm due to ‘unlucky’ reward realizations.

The Bayes-UCB algorithm of Kaufmann et al. [2012a] extends the UCB approach to the Bayesian setting. A posterior distribution over the expected reward of each arm is maintained, and at each step, the algorithm chooses the arm with the maximal posterior $(1 - \beta_t)$ -quantile, where β_t is of order $1/t$. Intuitively, using an upper quantile instead of the posterior mean serves the role of ‘optimism’, in the spirit of the original UCB approach. For the case of Bernoulli distributed outcomes and a uniform prior on the reward means, Kaufmann et al. [2012a] prove a *frequentist* upper bound on the expected regret that matches the lower bound of Lai and Robbins [1985].

3.3 Thompson Sampling

The Thompson Sampling algorithm (TS) suggests a natural Bayesian approach to the MAB problem using randomized probability matching [Thompson, 1933]. Let P_{post} denote the posterior distribution of

θ given the observations up to time t . In TS, at each time step t , we sample a parameter $\hat{\theta}$ from the posterior and select the optimal action with respect to the model defined by $\hat{\theta}$. Thus, effectively, we match the action selection probability to the posterior probability of each action being optimal. The outcome observation is then used to update the posterior P_{post} .

Algorithm 1 Thompson Sampling

```

1: TS( $P_{\text{prior}}$ )
   •  $P_{\text{prior}}$  prior distribution over  $\theta$ 
2:  $P_{\text{post}} := P_{\text{prior}}$ 
3: for  $t = 1, 2, \dots$  do
4:   Sample  $\hat{\theta}$  from  $P_{\text{post}}$ 
5:   Play arm  $a_t = \arg \max_{a \in \mathcal{A}} \mathbf{E}_{y \sim P_{\hat{\theta}}(\cdot|a)} [r(y)]$ 
6:   Observe outcome  $Y_t$  and update  $P_{\text{post}}$ 
7: end for

```

Recently, the TS algorithm has drawn considerable attention due to its state-of-the-art empirical performance [Scott, 2010, Chapelle and Li, 2011], which also led to its use in several industrial applications [Graepel et al., 2010, Tang et al., 2013]. We survey several theoretical studies that confirm TS is indeed a sound MAB method with state-of-the-art performance guarantees.

We mention that the TS idea is not limited to MAB problems, but can be seen as a general sampling technique for Bayesian learning, and has been applied also to contextual bandit and RL problems, among others [Strens, 2000, Osband et al., 2013, Abbasi-Yadkori and Szepesvari, 2015]. In this section, we present results for the MAB and contextual bandit settings, while the RL related results are given in Chapter 4.

Agrawal and Goyal [2012] presented *frequentist* regret bounds for TS. Their analysis is specific to Bernoulli arms with a uniform prior, but they show that by a clever modification of the algorithm it may also be applied to general arm distributions. Let $\Delta_a = \mathbf{E}_{\theta} [r(Y(a^*)) - r(Y(a))]$ denote the difference in expected reward between the optimal arm and arm a , when the parameter is θ .

Theorem 3.2. [Agrawal and Goyal, 2012] For the K -armed stochastic bandit problem, the TS algorithm has (frequentist) expected regret

$$\mathbf{E}_{\theta}[\mathbf{Regret}(T)] \leq \mathcal{O}\left(\left(\sum_{a \neq a^*} \frac{1}{\Delta_a^2}\right)^2 \ln T\right).$$

Improved upper bounds were later presented [Kaufmann et al., 2012b] for the specific case of Bernoulli arms with a uniform prior. These bounds are (order) optimal, in the sense that they match the lower bound of [Lai and Robbins, 1985]. Let $\mathbf{KL}_{\theta}(a, a^*)$ denote the Kullback-Leibler divergence between Bernoulli distributions with parameters $\mathbf{E}_{\theta}[r(Y(a))]$ and $\mathbf{E}_{\theta}[r(Y(a^*))]$.

Theorem 3.3. [Kaufmann et al., 2012b] For any $\epsilon > 0$, there exists a problem-dependent constant $C(\epsilon, \theta)$ such that the regret of TS satisfies

$$\mathbf{E}_{\theta}[\mathbf{Regret}(T)] \leq (1 + \epsilon) \sum_{a \neq a^*} \frac{\Delta_a (\ln(T) + \ln \ln(T))}{\mathbf{KL}_{\theta}(a, a^*)} + C(\epsilon, \theta).$$

More recently, Agrawal and Goyal [2013a] showed a *problem independent* frequentist regret bound of order $\mathcal{O}\left(\sqrt{KT \ln T}\right)$, for the case of Bernoulli arms with a Beta prior. This bound holds *for all* θ , and implies that a Bayesian regret bound with a similar order holds; up to the logarithmic factor, this bound is order-optimal.

Liu and Li [2015] investigated the importance of having an *informative* prior in TS. Consider the special case of two-arms and two-models, i.e., $K = 2$, and $\theta \in \{\theta_{\text{true}}, \theta_{\text{false}}\}$, and assume that θ_{true} is the true model parameter. When $P_{\text{prior}}(\theta_{\text{true}})$ is small, the *frequentist* regret of TS is upper-bounded by $\mathcal{O}\left(\sqrt{\frac{T}{P_{\text{prior}}(\theta_{\text{true}})}}\right)$, and when $P_{\text{prior}}(\theta_{\text{true}})$ is sufficiently large, the regret is upper-bounded by $\mathcal{O}\left(\sqrt{(1 - P_{\text{prior}}(\theta_{\text{true}}))T}\right)$ [Liu and Li, 2015]. For the special case of two-arms and two-models, regret lower-bounds of matching orders are also provided in Liu and Li [2015]. These bounds show that an informative prior, i.e., a large $P_{\text{prior}}(\theta_{\text{true}})$, significantly impacts the regret.

One appealing property of TS is its natural extension to cases with structure or dependence between the arms. For example, consider the following extension of the online shop domain: