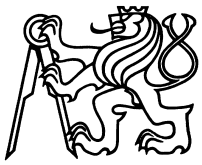
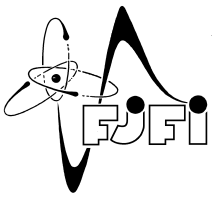


Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering



Real Options Valuation: A Dynamic Programming Approach

Oceňování projektů metodou reálných opcí z pohledu dynamického progamování

Master's Thesis

Author: **Filip Rolenec**

Supervisor: **Ing. Rudolf Kulhavý, DrSc.**

Language advisor: **Ing. Rudolf Kulhavý, DrSc.**

Academic year: 2020/2021

České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská

Katedra: matematiky Akademický rok: 2019/2020

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	Bc. Filip Rolenec
Studijní program:	Aplikace přírodních věd
Obor:	Matematické inženýrství
Název práce (česky):	Oceňování projektů metodou reálných opcí z pohledu dynamického programování
Název práce (anglicky):	Real Options Valuation: A Dynamic Programming Approach

Pokyny pro vypracování:

1. Seznamte se s tradičním přístupem k analýze reálných opcí obvyklým ve finanční analýze.
2. Formulujte analýzu reálných opcí jako úlohu stochastického řízení.
3. Navrhněte vhodnou metodu numerické aproximace dynamického programování.
4. Implementujte algoritmus oceňování ve Vámi zvoleném výpočetním nástroji a demonstrujte jeho chování na ilustrativní aplikaci a simulovaných datech.
5. Analyzujte přínosy teorie stochastického řízení pro analýzu reálných opcí. Identifikujte případná omezení a otevřené otázky.

Doporučená literatura:

1. Copeland, Thomas E., and Vladimir Antikarov, Real Options: A Practitioner's Guide. Revised ed. New York: Texere, 2003.
2. Guthrie, Graeme, Real Options in Theory and Practice. Oxford, England: Oxford University Press, 2009.
3. Powell, Warren B, Approximate Dynamic Programming: Solving the Curses of Dimensionality. 2nd ed. Hoboken, NJ: Wiley, 2011.
4. Puterman, Martin L., Markov Decision Processes: Discrete Stochastic Dynamic Programming. Hoboken, NJ: Wiley, 2005.
5. Vollert, Alexander. A Stochastic Control Framework for Real Options in Strategic Valuation. Boston, MA: Birkhäuser, 2003.

Jméno a pracoviště vedoucí diplomové práce:

Ing. Rudolf Kulhavý, DrSc.
Batličkova 253/1, , 182 00 Praha 8

Jméno a pracoviště konzultanta:

Datum zadání diplomové práce: 31.10.2019

Datum odevzdání diplomové práce: 4.5.2020

Doba platnosti zadání je dva roky od data zadání.

V Praze dne 11. října 2019

. . . . . .
garant oboru

. . . . . .
vedoucí katedry

. . . . . .
děkan

Acknowledgment:
I would like to thank my supervisor Ing. Rudolf Kulhavý, DrSc. for his professional guidance and all the advice given while creating this thesis.

Author's declaration:
I declare that this Master's thesis is entirely my own work and I have listed all the used sources in the bibliography.

Prague, February 23, 2020 Filip Rolenec

Název práce:

Oceňování projektů metodou reálných opcí z pohledu dynamického progamování

Autor: Filip Rolenec

Obor: Matematické inženýrství

Druh práce: Diplomová práce

Vedoucí práce: Ing. Rudolf Kulhavý, DrSc.

Abstrakt: Valuace převážné většiny investičních příležitostí se dnes stále určuje metodou diskontovaných peněžních toků (DCF) **??**. DCF metoda je přímočará a pro jednoduché projekty dává velmi přesné výsledky. Složitější projekty, které pro tuto práci definujme jako projekty s vysokou mírou vnitřní neurčitosti a existencí manažerských rozhodnutí značně ovlivňujících strukturu projektu (reálné opce), jsou dnes oceňovány metodou *real option analysis* (ROA). Metoda ROA přiznává hodnotu možnostem změny projektového plánu, v důsledku čehož je ohodnocení ROA vyšší než DCF.

Tato práce má za cíl interpretovat řízení projektů a s ním související zisk, jako řízení stochastického systému. Rozsáhlá teorie stochastického řízení **??**, **??**, **??**, dovoluje vybudovat novou teorii oceňování postavenoj na základech ROA, která mimo jiné řeší omezení ROA na pouze jediný zdroj neurčitosti - například cenu těžených minerálů na burze v budoucnosti **??**.

Nový originální přístup k oceňování projektů, který kombinuje dosavadní dosažené znalosti v ekonomické teorii (ROA) a teorii stochastického řízení systémů, je obecně definován v první části práce a prakticky otestován simulacemi na třídě problémů z oblasti těžebního průmyslu.

Klíčová slova: Analýza reálných opcí, Diskontované peněžní toky, Oceňování projektů, Stochastické řízení, Těžební průmysl

Title:

Real Options Valuation: A Dynamic Programming Approach

Author: Filip Rolenec

Abstract: The valuation of investment opportunities (projects) is nowadays still predominantly computed by the *discounted cash flow* (DCF) method **??**. DCF is straightforward and gives solid results for simple projects. For more complicated projects, which are in this thesis defined as projects with identifiable real options having substantial degree of inner uncertainty, the *real option analysis* (ROA) is now being used. ROA recognizes value in the ability to change projects' plans, which usually results in loss limitation, increasing the overall expected value of the project.

This thesis aims to interpret project execution and its valuation as a problem of stochastic decision control. The extensive stochastic decision control theory allows us, based on ROA, to design new valuation theory, that for example solves the limitation of ROA on only one source of uncertainty - such as the volatile price of the mined materials on the market.

New and original approach to project valuation, that combines both the economical theory (ROA) and the stochastic control theory is defined in the first part of the thesis and then tested by simulations on one class of valuation problems - valuation of projects in mining industry.

Keywords: Discounted cash flow, Mining industry, Project valuation, Real option analysis, Stochastic decision control

Contents

Chapter 1

Publications that I have read and short introduction to them

Books in the assignment description

- Real option, A practitioner's guide - Copeland, Antikarov 2003 **??**

Chapter 2

Introduction

In broad terms, this bachelor thesis studies decision making under uncertainty, one of the core parts of human activities. The process of decision making can be generally described as a sequence of actions taken on a system. Every performed action influences the system and generates a dependent output state. The structure of decision-making loop studied in the thesis is illustrated in Fig.**??**, while its parts are gradually clarified within the whole text. When the optimality of this sequence is studied, some measure of "profit" has to be introduced. This is usually done by assigning numerical value to each of the possible state-action-output state configurations. Since different actions lead to different output states it is reasonable to ask which sequence of actions leads to the highest profit on average. The mapping that generates such behaviour is called the optimal policy and its approximate design is the main topic of this work.

When studying real life systems, the output states tend to be uncertain, meaning that one action can lead to many states with generally different probabilities. Moreover, these *transition probabilities* are usually a priori unknown. The goal of statisticians is then to make the best estimates of the transition probabilities, based on the available data. These estimates can surely be made in various ways, but in this text we will talk mainly about the Bayesian approach to this problem, as it fits to the decision-making context.

As the research of decision-making theory can offer a significant improvement in various fields of human well-being, a lot of time and resources have been used for creating a coherent, widely acceptable and applicable theory. One of the key publications in this field is Puterman's book on Markov Decision Processes [9], which describes in detail the main mathematical framework that is used in this work. Another important book is Bellman's Dynamic Programming [**?**], which presents a simple but powerful approach for optimisation of the decision policy. The main idea in Bellman's theory, that is meant to be used primarily for finite-horizon processes, is that if the policy is to be optimal, its last action has to be. The theory of dynamic programming works well for systems with known parameters and it has been used in various fields of engineering, including, for instance, water engineering [12], speech recognition [10] or molecular biology [3]. A problem arises when the parameters are unknown or uncertain, which is almost always the case in real life applications. The need for estimation of the parameters based on the available data can be satisfied with a Bayesian approach, [8].

The Bayesian statistics paradigm as an alternative to the classical statistical model, offers an opportunity to implement an educated guess - a prior knowledge about the parameters of the process. The certainty about existence of a prior educated guess is well formulated by Peterka [8]: "No prior information is a fallacy: an ignorant has no problems to solve". It means that every time one is solving a decision problem, one has to have at least some information, or opinion, about how the system behaves, or what

its responses are expected to be. The practical usage of Bayesian estimation can be seen, for example, in works of Chaabane [7] (material engineering) or Bornignon [2] (petroleum engineering).

This text studies the optimal (in some sense) decision policy for discrete, finite-horizon Markov decision processes (MDPs), which form the backbone of the whole thesis. The optimal policy for MDPs can be designed by the theory of dynamic programming for known parameters. The policies for MDPs with unknown parameters can be also designed by the theory of dynamic programming, when outcomes of Bayesian estimation are included to the solution [1], but the actual feasibility of achieving the optimal policy is questionable. The exponential complexity that comes from the enormous increase in the size of the information state space is one of the main problem this work addresses.

Another problem, which is addressed in this work is the notorious exploration vs exploitation dichotomy [**?**]. It expresses the concern about estimation accuracy when only a limited amount of data is available and when actions have to be selected in parallel with the given exploitation aim. In other words, it is hard to assign an empirically supported probability to a process that has not happened.

The impact of the addition of deliberation cost into the model is the last aspect touched in this thesis. This term reflects the fact that "thinking cost resources", meaning that even the actions that could be described as "thinking" influence the profit of the decision making.

To solve all these problems generally would be very difficult, and that is why we have to limit our research to specific classes of MDPs. The first class can be characterized as MDPs that have fixed and finite number of actions and states. I call these MDPs *single systems*. The second class of MDPs, that is being studied, is called *multi-armed bandit*, even though it is actually a generalisation of the original definition of this term, [5]. The class of *multi-armed bandit* systems is a generalisation of the *single systems* in a sense that the former "consist of" several latter.

The structure of this thesis can be described as follows. A brief introduction to the mathematical model of MDPs and to the Bayesian statistics approach is presented in Chapter 2. Chapter 3 consists of both the formulations of the inspected optimisation problems and their solutions. Algorithms derived in Chapter 3 are then tested in simulations of various types of processes in Chapter 4, namely: *single systems* with known and unknown parameters. Chapter 5 provides a commentary on the conformity between the expected results from theory and the actual results from the simulations. Finally, Chapter 6 summarizes the work that has been done, underlines the main contributions contained in Chapters 3, 4 and 5, and offers suggestions for the future research.

Chapter 3

Preliminaries

This chapter introduces the mathematical theory and notions that are further used in this work. At first, several general mathematical definitions are presented. These are then followed by a specific notation for each part of the theory in respective sub-chapters.

In this work the usual notation for natural numbers $\mathbb{N}$, and real numbers $\mathbb{R}$ is used. The bold capital letters, such as $\mathbf{B}$, represent a set of all $b \in \mathbf{B}$, as in [11]. Cardinality of $\mathbf{B}$ is denoted as $|\mathbf{B}|$. In the whole work, $p(x)$ represents the probability of an event x. This $p(x)$ probability is mostly used for discrete-valued events, but it can also be in a form of probability density for the events with continuous range. Since this thesis works with conditional probability processes, expressing that probabilities of output states depend on actions, it uses conditional probability of an event x happening after the event y has already happened $p(x|y) = \frac{p(x,y)}{p(y)}$.

Also, the standard symbol for Kronecker delta function, $\delta(x, y)$ is used. This function is defined as $\delta(x, y) = 1$, if $x = y$ and 0, if $x \neq y$. For the parts of the thesis, where the complexity of algorithms is discussed, the Landau's O notion is used, [6].

Symbols that will be used in individual parts of this work originate mostly from three publications. The notion used for describing Markov's processes comes from [9], Bayesian statistics theory is mostly taken from [8], and finally the notion used for dynamic programming originates from [**?**].

3.1 Discrete Markov decisions processes

A lot of real life decision tasks can be approximately described by the mathematical framework called discrete Markov decision processes (MDPs), Fig. **??**. The definition of this mathematical framework is crucial for this work, as it is the basis upon which the theory of policy optimisation is built.

Definition 3.1. *Discrete Markov Decision Process $\mathcal{M}$ is defined as ordered set of five elements $(\boldsymbol{T}, \boldsymbol{S}, \boldsymbol{A}, P, R)$, where:*

- ***T** stands for a discrete, finite set of decision epochs; $\boldsymbol{T} = \{0, 1, 2, ..., |\mathbf{T}|\}$, where[1] $|\mathbf{T}| \in \mathbb{N}$*
- ***S** denotes a discrete, finite set of possible states of the system; $\boldsymbol{S} = \{1, 2, ..., |\mathbf{S}|\}$, where $|\mathbf{S}| \in \mathbb{N}$.*
- ***A** stands for a discrete, finite set of possible actions; $\boldsymbol{A} = \{1, 2, ..., |\mathbf{A}|\}$, where $|\mathbf{A}| \in \mathbb{N}$.*
- *P is a transition probability function, meaning that $P(\tilde{s}|a, s) \geq 0$, where $\sum_{\tilde{s} \in \mathbf{S}} P(\tilde{s}|a, s) = 1$, fulfilling the **Markovian feature** $P(\tilde{s}|a, s) = P(\tilde{s}|a, s, v)$, $\forall \tilde{s}, s \in \boldsymbol{S}$ and $\forall a \in \boldsymbol{A}$, where v represents any additional past observation.*

[1] Because the number of time epochs $|\mathbf{T}|$ is used frequently, the notation $|\mathbf{T}| = N$ is used.

- *R represents a real-valued reward function* $R = R(\tilde{s}, a, s)$*, where* $\tilde{s}, s \in \mathbf{S}$*,* $a \in \mathbf{A}$ *that is used for evaluation of the* $s \to a \to \tilde{s}$ *paths of the process.*

  The state in time epoch $t = 0$ *is known and it is denoted* s_0*.*

The special requirement for P called the Markovian feature means that the output in a MDP depends only on the previous state and the performed action. No other information has any influence on the probability of the next state.

3.2 Dynamic programming

When we talk about MDPs, it is usually because an optimisation task of some system behaviour is being solved. This means that mappings that assign actions to the observed states are mutually compared according to value functions, i.e. expected cumulative reward they induce. These mappings are called **policies** in this thesis[2], Definition 3.2, and the one that generates the most total reward on average is called the **optimal policy**. The optimal policy maximises the expected reward cumulated over all considered epochs.

Definition 3.2. *Let* $\mathcal{M} = (\boldsymbol{T}, \boldsymbol{S}, \boldsymbol{A}, P, R)$ *be a MDP, then a* ***policy*** $\pi_t \in \mathbf{\Pi_t}$*, on such* $\mathcal{M}$ *is defined* $\forall t \in \mathbf{T}$ *as a sequence of a decision rules* $(\pi_{t,\tau})_{\tau=t}^{N}$*.*

A ***decision rule*** $\pi_{t,\tau}$ *is the probability of selecting action a when the state s is observed, i.e.:* $\pi_{t,\tau}(a|s)) \geq 0, \sum_{a \in \mathbf{A}} \pi_{t,\tau}(a|s) = 1, \tau \geq t, \tau \in \mathbf{T}, \forall a \in \mathbf{A}, \forall s \in \mathbf{S}$*. When the decision rule* $\pi_{t,\tau}(a|s) = 1$ *for some action a and some state s, then we say that the decision rule is* ***deterministic****. A policy that consists only of the deterministic decision rules is called* ***deterministic policy****.*

The theory of dynamic programming, explained in detail in [**?**], is a simple yet powerful tool for obtaining the optimal policies in finite-horizon processes. As said in Introduction the idea is simple: *"if the policy is to be optimal the last decision has to be optimal"*. Before this idea can be mathematically formulated, the notion of value function is introduced.

Definition 3.3. *Let* $\mathcal{M} = (\boldsymbol{T}, \boldsymbol{S}, \boldsymbol{A}, P, R)$ *be a MDP and* $\pi_t \in \mathbf{\Pi_t}$ *a policy, then* ***value functions*** *of* $\mathcal{M}$*,* $\varphi_t^{\pi_t} : \mathbf{S} \to \mathbb{R}$ *are defined* $\forall \pi_t \in \mathbf{\Pi_t}$*,* $\forall t \in \mathbf{T} \setminus \{N\}$ *as:*

$$\varphi_t^{\pi_t}(s) = E\Big[\sum_{\tau > t, \tau \in \mathbf{T}} R(s_\tau, a_{\tau-1}, s_{\tau-1}) \Big| s_t = s \Big], \tag{3.1}$$

where $s_\tau, s_{\tau-1} \in \boldsymbol{S}$*, represent the state of the system in* τ*-th ((*τ*-1)-th) time epoch,* $s \in \mathbf{S}$ *stands for the state for which the value function is evaluated and* $a_{\tau-1}$ *represents the action taken in the* $(\tau - 1)$*-th time epoch. The symbol E represents the conditional expectation.*

The term optimal value function $\varphi_t^o = \max_{\pi_t \in \mathbf{\Pi_t}} \varphi_t^{\pi_t}$ *is further used as a value function of the optimal policy* π_t^o*, i.e. a policy maximising the value function.*

This value function $\varphi_t^{\pi_t}(s)$ thus represents the expected cumulative reward from the state $s \in \mathbf{S}$ at the time epoch $t \in \mathbf{T}$ for the policy π_t.

Now, when the optimal policy is desired, one can transform the problem to finding the optimal value function and then simply declare the optimal policy as the argument of the maxima.

When we would like to obtain the optimal value function, we could first think about the simple maximisation of $\varphi_t^{\pi_t}$ over the set $\mathbf{\Pi_t}$. But even if we limited our maximisation to the purely deterministic

[2] It is worth noting that a term *optimal decision strategy* is used for optimal policies in some other works, for example [**?**].

policies, it would still mean to compare $|\mathbf{A}|^{|\mathbf{S}|(N-t)}$ distinguishable ones on $|\mathbf{S}|$ states, $|\mathbf{A}|$ actions, in $N-t$ time epochs. This means that the algorithm would still have exponential complexity, which is generally considered infeasible. Thus another algorithm for obtaining the optimal value function has to be used. It is the dynamic programming algorithm that offers an alternative approach for evaluating the optimal value function φ_t^o with a significantly lower complexity. Its main idea is that the φ_t^o is computed layer by layer as it can be seen in Theorem 1.

Theorem 1. *Let $\mathcal{M} = (\boldsymbol{T}, \boldsymbol{S}, \boldsymbol{A}, P, R)$ be a MDP, then the optimal value function $\varphi_t^o(s) = \max_{\pi_t \in \mathbf{\Pi_t}} \varphi_t^{\pi_t}$ can be computed recursively $\forall t \in \mathbf{T} \setminus \{N\}$ through the equation:*

$$\varphi_t^o(s) = \max_{a_t \in \mathbf{A}} E\Big[R(s_{t+1}, a, s) + \varphi_{t+1}^o(s_{t+1}) | a_t, s\Big], \tag{3.2}$$

considering that $\varphi_N^o = 0$ as Definition 3.3 implies. Furthermore the optimal policy π_t^o, as the argument of maxima, is concentrated on maximising arguments in Equation (3.2).

Proof. This theorem will be proven via a finite backward mathematical induction.

At first, let us take an action a_{N-1}^o in the time epoch $N-1$ defined as:

$$a_{N-1}^o(s) = \operatorname{Arg}\max_{a \in \mathbf{A}} E[R(s_N, a, s)|a, s]. \tag{3.3}$$

By the definition of maxima the inequality

$$E[R(s_N, a, s)|a_{N-1}^o(s), s] \geq E[R(s_N, a, s)|a, s] \tag{3.4}$$

holds $\forall a \in \mathbf{A}$. Now, let us take any decision rule $\pi_{N-1,N-1}$. This function consists only of a single decision rule, generating the action $a \in \mathbf{A}$ given by $\pi_{N-1,N-1}(a|s) \geq 0$. By multiplying the inequality by these non-negative numbers and summing over all $a \in \mathbf{A}$ we get

$$E[R(s_N, a, s)|a_{N-1}^o(s), s] \geq E_{\pi_{N-1,N-1}}[R(s_N, a, s)|s], \tag{3.5}$$

where $E_{\pi_{N-1,N-1}}$ represents a mean value based on the randomised decision rule $\pi_{N-1,N-1}$. By defining $\pi_{N-1,N-1}^o(a|s) = \delta(a_{N-1}^o(s), a)$ the left side of the inequality can be rewritten as

$$E_{\pi_{N-1,N-1}^o}[R(s_N, a, s)|s] \geq E_{\pi_{N-1,N-1}}[R(s_N, a, s)|s]. \tag{3.6}$$

Inequality (3.6) shows both, the optimality of the deterministic policy $\pi_{N-1,N-1}^o$ and the validity of Equation (3.2) for $t = N-1$, since $\varphi_N^o(s) = 0\ \forall s \in \mathbf{S}$.

The proof for the remaining induction steps is identical to the one presented above, with the exception that instead of function R, the sum $R + \varphi_\tau^o$ is used, when the validity for (3.2) is being proven for $t = \tau - 1$. □

Now it can be seen how the optimal value functions φ_t^o are computed in the dynamic programming algorithm. At first the value $\varphi_N^o(s)$ is evaluated $\forall s \in \mathbf{S}$ from Definition 3.3. This is simply 0, as it is a maximum of zero sums $\forall s \in \mathbf{S}$. Theorem 1 is then used for the first time for evaluating the optimal value function in the time epoch $t = N-1$, $\varphi_{N-1}^o(s)$, $\forall s \in \mathbf{S}$. Equation (3.2) is then recursively used for lower and lower levels until the values of $\varphi_0^o(s)$ are obtained $\forall s \in \mathbf{S}$. The optimal deterministic policy is concentrated on maximising arguments found during evaluations. [3]

It can also be seen that the complexity, in this case the number of operations needed, of this algorithm is $O(N|\mathbf{S}|^2|\mathbf{A}|)$. There are N optimal value functions φ_t^o, one for each time layer, each with $|\mathbf{S}|$ values needed to be computed. To evaluate one value means to compare average rewards from $|\mathbf{A}|$ actions, where one average reward takes $O(|\mathbf{S}|)$ operations to obtain.

[3] Okay like this?

3.3 Bayesian estimation

This section briefly recalls the Bayesian approach to the statistical decision theory for inspected MDPs with an unknown parameter θ.

As this thesis studies the optimal policies for MDPs with unknown transition probability function $P(\tilde{s}|a, s)$ there is a need for its estimation. At first the function P is parametrized by the parameter $\theta \in \mathbf{\Theta}$. This dependency is described as $P(\tilde{s}|a, s, \theta)$. This reduces estimation of P to estimation of eh parameter θ.

The estimation of the parameter θ, which is used for determining the optimal policy, can be logically based only on the previously measured data, which are in the considered case compressed into a sufficient statistic denoted as $v = v(\tilde{s}|a, s) \in \mathbf{V}$. The values of the v statistic $v(\tilde{s}|a, s)$, where $\tilde{s}, s \in \mathbf{S}, a \in \mathbf{A}$, represent the number of times the path $s \to a \to \tilde{s}$ has been measured through the history of such MDP.

The estimation can be done by classic approach or the alternative Bayesian where both assume to know the family of the distribution of data, $\mathscr{F} = \{P(\tilde{s}|a, s, \theta)|\tilde{s}, s \in \mathbf{S}, a \in \mathbf{A}, \theta \in \mathbf{\Theta}\}$, parametrized by finite-dimensional $\theta \in \mathbf{\Theta}$. The main advantage of the Bayesian estimation is that it provides the needed system model even with no or small amount of data forming v available. This feature of the Bayesian approach is needed in order to perform dynamic programming Section 3.2 and it is the main reason why it is used in this work.

The main feature of the Bayesian statistics is that it treats the unknown parameter θ as random variable. A distribution of this random variable" θ is then studied.

Before any measurement is made, a statistician has to provide a probability density function of continuous-valued θ, called the prior probability $p_0(\theta)$, which should express some knowledge about process that is being studied.[4] This probability then serves as a starting point, which is being adjusted by the truly measured data represented by the statistic v. The prior probability $p_0(\theta)$ can be derived in several ways [**?**] by exploiting

- an information from the past, data from the previous inspection of the same (or similar) problems;
- a subjective educated guess of the statistician or an expert on the topic;
- a combination of the former two, for instance by using their weighted average;
- the principle of insufficient reasons, where the $p_0(\theta)$ is uniform distribution on the domain of θ, meaning that θ has the same probability of having any value possible.

The principle of insufficient reasons can be used even when the estimated parameter is from set that has an infinite measure. Then $p_0(\theta)$ is not an actual probability distribution (as it does not integrate to 1) but rather a improper distribution, with which is worked in the same way as with the ordinary probability distribution. A detailed construction of $p_0(\theta)$ can be found in [8].

Now, when the *prior probability* $p_0(\theta)$ is known, it can be adjusted for the actually measured data represented by the statistic v and then used for the prediction of the future behaviour of the system that is needed in dynamic programming. The adjustment is represented by an equation that comes from more general identity called the Bayes rule, Theorem 2, [11].

Theorem 2. *Let x and y be random variables and $\boldsymbol{X}$ a set of all possible values of the variable x. Then the conditional probability $p(x|y)$ fulfils the following equality:*

$$p(x|y) = \frac{p(y|x)p(x)}{\int_{\boldsymbol{X}} p(y|x)p(x)dx}. \tag{3.7}$$

[4] This is new

When identifying $x = \theta$, $\mathbf{X} = \mathbf{\Theta}$, and $y = v$, Equation (3.7) yields:

$$p(\theta|v) = \frac{p(v|\theta)p_0(\theta)}{\int_{\mathbf{\Theta}} p(v|\theta)p_0(\theta)d\Theta}, \tag{3.8}$$

where $p(\theta|v)$ is called a *posterior probability* of the parameter θ based on the value of statistic v. For a given parametric model, $P(\tilde{s}|a, s, \theta)$ and policies, for which θ is unknown, i.e. $\pi_{t,\tau}(a|s, v, \theta) = \pi_{t,\tau}(a|s, v)$, see *Natural conditions of control* [8], the Bayes rule provides the update of the statistic v on $\tilde{v}$ by using observed $s, a, \tilde{s}$ as

$$p(\theta|\tilde{v}) = \frac{P(\tilde{s}|a, s, \theta)p(\theta|v)}{\int_{\mathbf{\Theta}} P(\tilde{s}|a, s, \theta)p(\theta|v)d\theta}. \tag{3.9}$$

Now, we are able to derive the estimation of the parameter θ based on the statistic v according to Equation (3.3). However the main question about the future behaviour of the process that is described by $P(\tilde{s}|a, s, \theta)$ remains. Our incomplete knowledge of the parameter θ extends the state s on a generalised state (s, v), where v is a sufficient statistic for θ estimation and is called the information state.

Prediction[5] of the s-part of the generalised state then reads

$$P(\tilde{s}|a, s, v) = \int_{\mathbf{\Theta}} P(\tilde{s}|a, s, \theta)p(\theta|v)d\theta, \tag{3.10}$$

whereas the evolution of v to $\tilde{v}$ is given by (3.9). It reflects that system and knowledge accumulation dynamically evolve in dependence on chosen action and that the generalised state (s, v) evolves in Markovian way.

[5]Common name for the transition probability gained by learning.

Chapter 4

Addressed tasks and their solutions

This chapter delimits the optimisation problems of chosen classes of MDPs and provides their theoretical solutions. Each sub-chapter concentrates on one class of MDPs, starting with the *single system* for known and unknown transition probability function P in Sections 4.1 and 4.2, which are then followed by optimisation of the *multi-armed bandit* in Section 4.3.

It is shown that when using the dynamic programming algorithm for obtaining the optimal policies for the processes with known transition parameters P, the solution is straightforward and not hard to apply. On the other hand, when the solution to the optimisation problems for systems with unknown P is desired, one cannot offer a feasible and unambiguous one. I present three types of algorithms, which are all based on the combination of the dynamic programming and the Bayesian estimation. The difference in these is in exploiting the estimation results, which has a significant influence on the overall complexity.

4.1 Optimal decision policy for a single system with known model

The first goal of this thesis is to find the optimal decision policy of MDP that I call *single system* if the transition probability function P is known. This MDP is defined as follows.

Definition 4.1. *Let $\mathcal{M} = (\boldsymbol{T}, \boldsymbol{S}, \boldsymbol{A}, P, R)$ be a MDP. Then $\mathcal{M}$ is called **single system**, when the sets* $\mathbf{A}$ *and* $\mathbf{S}$ *can be represented as $A = \{1, 2, ..., |\mathbf{A}|\}$, or $S = \{1, 2, ..., |\mathbf{S}|\}$ respectively.*

These types of processes correspond with the idea that a system can be in one of finite number of states and then after "turning on" generates an output, which is an observable state of the system. This output state is rewarded based on the initial state and performed action via the reward function R. The actions can the be interpreted as "putting the system in a different state".

Example 1. *A real life example of the single system MDP is an evaluated biased coin tossing. Imagine a game where you are tossing a coin and you are rewarded with* \$1 *every time you get Heads as the output. The coin bias is known and the result of the toss depends on the side, from which the coin is tossed. The action of turning the coin after its landing (and before another toss) is penalised by paying a dime. The question is how to behave, what actions to perform, in each time epoch of this process when you want to get as much money as you can in a finite, N-round game.*

Finding the optimal policy for *single system* MDP $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \mathbf{A}, P, R)$ with known transition probability function P is the easiest problem that is addressed in this thesis. The solution in form of a decision table could come straight from definition of the *value function*, Definition 3.3. The optimal policy π_0^o for the initial state $s_0 \in \mathbf{S}$ would be the argument of the maxima of the optimal value function $\varphi_0^o(s_0)$.

Such an algorithm would have to evaluate the expected reward for all the possible $|\mathbf{A}|^{|\mathbf{S}|N}$ distinguishable deterministic policies while finding the maximum of them. This algorithm could be categorized as a brute force algorithm and the computational complexity would be at least exponential in a number of operations.

Much more efficient Algorithm 1, which is based on Theorem 1, is used in this thesis. Theorem 1 offers to recursively evaluate all the optimal value functions φ_t^o with the complexity $O((N-t)|\mathbf{S}|^2|\mathbf{A}|)$. This means that the optimal policy π_0^o for the *single system* MDP class can be obtained as the argument of the optimal value function φ_0^o with a complexity $O(N|\mathbf{S}|^2|\mathbf{A}|)$.

We can now describe the dynamic programming algorithm in detail.

Algorithm 1 Finding the optimal policy for a *single system* MDP with known P

Require: $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \mathbf{A}, P, R)$
1: $\varphi_N^o(s) \leftarrow 0, \forall s \in \mathbf{S}$ ▷ Based on Definition 3.3.
2: $t \leftarrow N$
3: **while** $t \neq 0$ **do**
4: **for** each $s \in \{1, 2, ..., |\mathbf{S}|\}$ **do**
5: $\varphi_{t-1}^o(s) \leftarrow$ *Equation* (3.2) ▷ With known P and φ_t^o
6: $t \leftarrow t - 1$
7: $\pi_0^o(s) \leftarrow argmax\ \varphi_0^o(s)\ \forall s \in \mathbf{S}$, ▷ Deriving the optimal policy
8: **return** π_0^o

This algorithm offers not only the explicit optimal policy in terms of a decision table but also the expected cumulative reward that is to be gained from the beginning in each state $s \in \mathbf{S}$ when the optimal policy is used through the process. This is expressed through the value $\varphi_0^o(s)$. With the knowledge of both π_0^o and the values $\varphi_0^o(s)$, a simulation of the actual decision making on $\mathcal{M}$ can be performed and the experimental cumulative rewards can be compared to the expected optimal ones. This simulation can be found in Section 5.1.

4.2 Optimal decision policy for a single system with unknown model

The second optimisation problem this thesis copes with has almost identical setting as the first one. The only but crucial difference from the first problem presented in the previous section is that the decision maker lacks information about the transition probability function P.

It is reasonable to expect that this lack of information makes the conclusions about the optimal policy more difficult to make and that some estimations of P have to be made. These estimates, each in time $t \in \mathbf{T}$, are noted as P_t and they are called predictors. [1] The adopted Bayesian approach expresses the incomplete knowledge by considering family $\mathscr{F}$ parametric models. It updates sufficient statistics $v_t \in \mathbf{V}_t$ determining posterior probabilities of $\theta \in \mathbf{\Theta}$. Then, it evaluates predictors P_t, which relate the observed past states and the selected actions to future states. Thus, the predictor serves as the known system model needed in the design of policy optimal under the available information.

The v_t is called information state and it consists of the frequencies of paths $s \rightarrow a \rightarrow \tilde{s}$ that have been measured until the time epoch t, see Section 4.2.1. The $\mathbf{V_t}$ is the set of all the possible information states.

Now a question how to determine the optimal action in time epoch t based on the available information arises. When the theory of dynamic programming is to be used, one have to generalise the definition

[1] Please read the following paragraph, it is new

of a "state" from $s \in \mathbf{S}$ to a (s, v_t) pair because the predictor P_t modelling the system depends on v_t. This exact dynamic programming approach is then shown to be infeasible due to the extreme increase in the cardinality of $\mathbf{V_t}$ with higher t. This approach is presented in 4.2.2.

The extreme complexity caused by incomplete knowledge is avoided by "freezing" the evolution of the informational state v_t used for dynamic programming. Moreover, the predictor $P(s_{\tau+1}|a_\tau, s_\tau, \hat{\theta}_t)$ is used in optimisation for $\tau \geq t, \tau \in \mathbf{T}$, where $\hat{\theta}_t$ is the current (in time epoch t) point estimate of θ, see Section 4.2.3. This approximation approach is called certainty equivalence, as it declares the estimate $\hat{\theta}_t$ to be equivalent to the actual parameter θ. [2]

Now two algorithms are available, one more precise but infeasible, and the other one which has significantly lower complexity, but it is sub-optimal as it does not fully exploit all available information.

One could wonder if there is an algorithm "in between". Some algorithm providing the best results (closest to the ones in Section 4.2.2) while respecting the amount of computing power available. The answer to this question can be found in Section 4.2.4 where the m-Step approximation algorithm is presented. This m-Step approximation algorithm works as a hybrid of the two formerly derived ones. At first, the CE approach is used for the evaluation of the estimate of the value function in time epochs $\tau \geq t + m$. Then the exact dynamic programming is initiated for the computation of the estimate of the optimal value function for the remaining $\max\{m, N - t\}$ time epochs. A small m is used so that the statistician using this algorithm is able to modify the feasibility of this algorithm with respect to the available computing power. This algorithm is presented in 4.2.4.[3]

4.2.1 Bayesian estimation of transition probability function P

For the computation of the Bayesian estimate of the transition probability function, two entities are needed. The family of the distribution of $P(\tilde{s}|a, s, \theta)$, $\mathscr{F}$, and the prior probability of θ, $p_0(\theta)$. These are then used for the evaluation of the Bayesian estimate of θ, called θ_t, which provides the P_t estimation through Equation (3.10), and are defined as follows.

In the considered discrete-valued case, transition probability functions are finite-dimensional tables. Thus, they can be parametrised by probability values as:

$$\mathscr{F} = \left\{P(\tilde{s}|a, s, \theta)\middle|\theta(\tilde{s}|a, s) = \prod_{\tilde{s}', s' \in \mathbf{S}} \prod_{a' \in \mathbf{A}} \theta(\tilde{s}'|a', s')^{\delta(\tilde{s}', \tilde{s})\delta(s', s)\delta(a', a)}\right\}, \tag{4.1}$$

where the set of parameters θ is determined by their meaning. They have to be non-negative and sum over $\tilde{s}$ one must be the identity matrix. This seemingly complex expression of $\mathscr{F}$ members hints how to choose the prior probability $p_0(\theta)$. It is determined by parameters $w = w(\tilde{s}|a, s) \geq 0$,[4] where $\tilde{s}, s \in \mathbf{S}, a \in \mathbf{A}$, as:

$$p_0(\theta|w) \propto \prod_{\tilde{s}, a, s} \theta(\tilde{s}|a, s)^{w(\tilde{s}|a,s)-1}, \tag{4.2}$$

where $\propto$ means proportionality. The normalising factor is a product of multivariate beta functions

$$\prod_{s,a} \frac{\prod_{\tilde{s}} \Gamma(w(\tilde{s}|a, s))}{\Gamma(\sum_{\tilde{s}} w(\tilde{s}|a, s))}. \tag{4.3}$$

The Bayesian rule, Equation (3.7), point-wise multiplies prior probability by likelihood, i.e. by members of $\mathscr{F}$ evaluated at observed data. The "complex" form of $\mathscr{F}$, Equation (4.1), immediately reveals the form of sufficient statistics $v_t = v(\tilde{s}|a, s), \tilde{s}, s \in \mathbf{S}, a \in \mathbf{A}$ as the number of how many times the

[2] Check this again please, I have made hats for the point estimates and referenced the solution. And overall changed it a bit

[3] This paragraph is totally new, I have changed a lot about what you have suggested, hope is it okay still.

[4] Please check the whole section up to this, huge changes were made

transition $s \to a \to \tilde{s}$ occurred until the time epoch t. This also implies that the posterior probability has the same functional form as the prior probability. This reduces the updating of the posterior probability into simple updating of counts represented in informational state v_t. It also interprets the parameter $w(\tilde{s}|a, s)$ determining prior probability as a variable that represents the number of paths already measured by the previous experiment. [5]

Now when the prior probability $p_0(\theta)$, the family of of the P distribution, $\mathscr{F}$, and the reason for their choice was presented, we can see how the actual posterior estimation of P looks like. This estimate is represented by predictors P_t which are computed based on the measured statistic v_t according to Equation (3.3). The result of the integration in this equation is

$$P_t(\tilde{s}|a, s, w, v_t) = \frac{w(\tilde{s}|a, s) + v(\tilde{s}|a, s)}{\sum_{s' \in \mathbf{S}}(w(s'|a, s) + v(s'|a, s))}, \tag{4.4}$$

where the detail derivation of this equation can be found in [**?**].

4.2.2 Optimal Bayesian decision policy

The first algorithm for obtaining a sub-optimal policy for a *single system* with unknown transition probability function P is called the optimal Bayesian algorithm, Algorithm 2. It is again based on the theory of dynamic programming explained in Section 3.2.

The actual function P needed for the evaluation of the mean values in all time epochs $t \in \mathbf{T}$, which are crucial for the dynamic programming, is replaced by the Bayesian predictor P_t. This predictor is constructed according to the Section 4.2.1 based on the informational state v_t, which describes the history of the process.

The addition of the information state v just slightly transforms the formulation of the important equation (3.2) derived in Theorem 1. If we understand the new couple (s, v) as a generalised state, then Theorem 1 says that the optimal value function $\varphi_t^o(s, v)$ can be computed $\forall s \in \mathbf{S}$, $\forall v \in \mathbf{V_t}$ as:

$$\varphi_t^o(s, v) = \max_{\pi_t \in \mathbf{\Pi_t}} E\Big[R(\tilde{s}, a_t, s) + \varphi_{t+1}^o(\tilde{s}, \tilde{v})|s, v\Big], \tag{4.5}$$

where $\tilde{s} \in \mathbf{S}$ and $\tilde{v} \in \mathbf{V_{t+1}}$.

The only thing that has to be taken into account is that the expected value E is based on particular P_t determined by the history $v_t \in \mathbf{V_t}$ and the prior information $w \in \mathbf{W}$.

It is worth noting that some generalised states $(\tilde{s}, \tilde{v}) \in (\mathbf{S}, \mathbf{V})$, in fact most of them, can be obtained with zero probability, due to the evolution of the information part of the generalised state. This significantly lowers the complexity of the actual implementation of the algorithm.

Since all the paths are taken into account with this approach, one can compute the optimal value functions $\varphi_t^o(s, v)$, $\forall s \in \mathbf{S}$ and $\forall v \in \mathbf{V_t}$, in the same way as earlier, with the difference of dependence on the predictor P_t given by v_t.

The optimal policy is now computed for all possible paths of the process that can occur. That means that each time we are faced with a decision, we already know what the optimal one is. It is the argument of the maxima of function $\varphi_t^o(s, v_t)$ provided that we are dealing with a state $s \in \mathbf{S}$ in time epoch $t \in \mathbf{T}$ when the history described by statistics $v_t \in \mathbf{V_t}$ has happened.

The algorithm of obtaining the optimal policy is similar to the one for known parameters and it is described in detail as Algorithm 2.

When we look at the algorithm in detail it can be seen that the computational complexity will generally be extreme for this type of algorithms. Only the computation of the pre-last optimal value function

[5] Also a brand new paragraph, please check this one also.

Algorithm 2 Optimal Bayesian algorithm for the optimisation of *single system* MDP with unknown P

Require: $\mathbf{T}, \mathbf{S}, \mathbf{A}, R, w, \mathscr{F}, s_0, P$ unknown.

$\varphi_N^o(s, v_N) \leftarrow 0, \forall s \in \mathbf{S}, \forall v_N \in \mathbf{V_N}$ ▷ As no further gain is expected

$t \leftarrow N$

while $t \neq 0$ **do**

  for each $(s, v_t) \in (\mathbf{S}, \mathbf{V_t})$ **do**

    $\varphi_{t-1}^o(s, v_{t-1}) \leftarrow$ Equation (4.5) ▷ Considering the P_t based on v_t

    $t \leftarrow t - 1$

    $\pi_0^o(s, v_0) \leftarrow argmax\ \varphi_0^o(s, v_0)$

return π_0^o

$\varphi_{N-1}^o(s, v_{N-1})$ means to compute and store the results for $|\mathbf{S}||\mathbf{V_{N-1}}|$ variables, where the $|\mathbf{V_{N-1}}|$ is the number of all the possible paths the process could have gone through until the time epoch $N - 1$.

Lets derive this number to show how many values have to be computed, for example, for the mentioned pre-last optimal value function φ_{N-1}^o. The problem of finding the cardinality $|\mathbf{V_{N-1}}|$ can be interpreted as giving a certain amount of marbles into another number of pockets, where the pockets represent the distinguishable paths $s \to a \to \tilde{s}$, and the marbles represent the path $s \to a \to \tilde{s}$ in each time epoch. When a path $s \to a \to \tilde{s}$ is measured, its pocket gets a marble. The combinatorics theory, which can be found in [4], says that the number of such possible "marble distributions" is $\binom{|\mathbf{S}||\mathbf{A}||\mathbf{S}|+N-2}{N-1}$, which is for example for the settings in Section 5.2.2, where $|\mathbf{S}| = 3 = |\mathbf{A}|$ and $N = 30$, number 3.56×10^{15}.

Because of the enormous requirements for computing power and storage, determined by the cardinality of the information state space, this algorithm is declared as infeasible and an alternative approach is offered.

4.2.3 Certainty equivalence

In this section a second, now feasible, algorithm that copes with the optimisation problem of *single system* MDP with unknown transition probability function P is presented. Because the policy derived by this algorithm is based on the CE approximation it is not generally the optimal one. However, the derived policy is the best one based on this type of approximation and that is why we call it the sub-optimal policy.

As said in the end of Section 4.2.2 the main reason that the first algorithm fails in actual computation, is that the addition of the information state to the evaluation extremely increases the complexity of the algorithm. The second algorithm is presented with a goal to eradicate this dependency on the information state. The theory of certainty equivalence (CE) offers such eradication.

The need for informational state v arose from the Bayesian theory which says that the predictor P_t depends on the history of the MDP until the time epoch t, through the distribution of a random variable θ_t. [6] The theory of CE freezes the evolution of the sufficient statistic and uses the point estimate θ_t of θ instead of the unknown θ. As a result, the estimate θ_t of θ does not change. We define the point estimate θ_t as the expectd value

$$\hat{\theta}_t = E[\theta|v], \tag{4.6}$$

[6] In my opinion the θ_t notion should stay, the predictor does depend on the history through the distribution of the theta estimate, not theta itself

[7]where $v \in \mathbf{V_t}$. The used estimate of P, denoted as $\hat{P}$ is time invariant during optimisation and it is the main contribution of the CE approach.

It is worth noting that the estimate $\hat{P}_t$ is identical to the Bayesian estimate determined by the equation (4.4). This is not a coincidence but a result of our choice of θ parametrisation and the choice of $E[\theta|v]$ as our point estimate of θ.

This approach makes the information state redundant in dynamic programming because we can now use the dynamic programming algorithm for these point estimates $\hat{P}_t$ with no further complications. The problem was now reduced to finding the optimal policy for the *single system* with known parameters. The CE algorithm for *single system* MDPs is in detail described as Algorithm 3.

Algorithm 3 CE algorithm for the optimisation of *single system* MDP with unknown P

Require: $\mathbf{T}, \mathbf{S}, \mathbf{A}, R, w, \mathscr{F}, s_0, P$ unknown.

1: Make the first point estimate $\hat{P}_0$ based only on w. ▷ Equation (4.4)
2: **for** $i = 0, i \leq N, i++$ **do**
3: Find the optimal policy $\pi_0^o(s_0)$ for $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \mathbf{A}, \hat{P}_i, R)$ through Algorithm 1.
4: Perform the first action of the derived policy and observe the output state.
5: $s_0 \leftarrow$ output state.
6: Re-evaluate the point estimate $\hat{P}_i$ to $\hat{P_{i+1}}$ based on the state-action-output state triplet. ▷ Equation (4.4)
7: $\mathbf{T} \leftarrow \mathbf{T} \setminus \{|\mathbf{T}|\}$
8: The sub-optimal policy is defined by the performed actions through the process.

The simulation, where a policy derived by this algorithm is used is presented in the Section 5.2.1.

4.2.4 m-Step approximation

The last presented algorithm, Algorithm 4, offers to balance the quality of the derived sub-optimal policy with the computing power needed for its evaluation. When working with this algorithm the statistician controls the complexity based on parameter m, which is used in the name of the m-Step approximation algorithm. This parameter m represents for how many time epochs is the dynamic programming performed "properly", meaning with the information state in mind, and after what amount of time epochs the CE approximation is used.

In the first algorithm, P is taken as a random variable for the whole process. In the second algorithm this random variable P is replaced by the point estimates $\hat{P}_t$. Now, the transition probability function P is considered to be a random variable for the first m steps of the process, after which it is substituted by its point estimate $\hat{P}_t$. The detailed description can be seen in Algorithm 4.

The simulation, where a policy derived by this algorithm is used is presented in the Section 5.2.2.

4.3 Optimal decision policy for multi-armed bandit

This section presents the definition and the solution to the decision-making problem on a second class of MDPs. The members of this class are presented as consisting of [8]several *single systems* defined in Section 4.1. This class is called *multi-armed bandit* MDPs and it is defined in detail as follows.

[7]I have eliminated the t dependency on the right side, but I want to keep the hat on the left side.
[8]This is new, please check.

Algorithm 4 m-Step algorithm for the optimisation of *single system* MDP with unknown P

Require: $\mathbf{T}, \mathbf{S}, \mathbf{A}, R, w, \mathscr{F}, s_0, P$ unknown.

Make the point estimate $\hat{P}_0$ from the CE theory. ▷ Equation (4.4)
for $i = 0, i \leq N, i++$ **do**
  $\varphi_N^o(s) \leftarrow 0, \forall s \in \mathbf{S}$. ▷ Based on Definition 3.3.
  $t = N - i$.
  while $t > m$ **do**
    for each $s \in \mathbf{S}$ **do**
      $\varphi_t^o(s)$ is evaluated based on Algorithm 1 used for $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \mathbf{A}, \hat{P}_i, R)$.
      $t \leftarrow t - 1$.
  while $t \neq 0$ **do**
    evaluate $\varphi_t^o(s, v)$ from the $\varphi_{m+1}^o(s)$, taken as the horizon value $\varphi_{m+1}^o(s, v)$ $\forall v \in \mathbf{V_{t+1}}$ similarly to Algorithm 2.
    $t \leftarrow t - 1$.
  $\pi_0^o(s_0) \leftarrow argmax\varphi_0^o(s_0, v)$ $\forall s \in \mathbf{S}$. ▷ $v \in \mathbf{V_0} = \{0\}$
  The action advised by $\pi_0^o(s_0)$ is taken and the output state $\tilde{s}$ is measured.
  $s_0 \leftarrow \tilde{s}$.
  The point estimate $\hat{P}_i$ is adjusted to $\hat{P_{i+1}}$, based on the information of the measured $(\tilde{s}, a, s)$ triplet. ▷ Via Equation (4.4).
  $\mathbf{T} \leftarrow \mathbf{T} \setminus \{|\mathbf{T}|\}$
The sub-optimal derived policy π_0^o is defined through the actions that are taken through the process.

Definition 4.2. *Let $\mathcal{M} = (\boldsymbol{T}, \boldsymbol{S}, \boldsymbol{A}, P, R)$ be a MDP. Then $\mathcal{M}$ is called a **multi-armed bandit** MDP if the structure of sets $\mathbf{A}$, $\mathbf{S}$ and functions P and R fulfils the following requirements.*

- $\mathbf{S}$ *can be represented as* $\mathbf{S} = \{(s^a|s^p)|\ s^a \in \mathbf{S_a}, s^p \in \mathbf{S_1} \times \cdots \times \mathbf{S_{|K|}}\}$, *where* $|\mathbf{K}| \in \mathbb{N}$ *is the cardinality of an index set* $\mathbf{K}$ *and* $\mathbf{S_k}$, $k \in \{1, 2, ..|\mathbf{K}|\}$, *are the state sets of the respective single systems. The number s^a is the number of an "active" single system. The $|\mathbf{K}|$-dimensional vector s^p is called the positional state and it represents the "positions" of each of the considered $|\mathbf{K}|$ single systems.*

- $\mathbf{A}$ *can be represented as* $\mathbf{A} = \{(a^a, a^p)|\ a^a \in \{1, 2, ..., |\mathbf{K}|\}, a^p \in \{1, 2, ..., |\mathbf{A_{a^a}}|\}$[9], *where* $|\mathbf{A_{a^a}}| \in \mathbb{N}$ *is the cardinality of action set* $\mathbf{A_{a^a}}$ *of the $a^a - th$ single system. The first part of the action a_t^a makes the $a-th$ system active in the time epoch t, whereas the second part a^p influences its position state, the $a - th$ dimension of the positional state s^p.*

- *The function $P(\tilde{s}|a, s)$ keeps state entries unchanged for non-active systems and for the active system it has its own transition probability evolving its positional state in the same way it would when the other single system were not considered.*

- *Values $R(\tilde{s}, a, s)$ are zero for non-active systems and the active system is rewarded on its own, again as if the other single systems did not exist. On the top of this evaluation the deliberation cost is introduced, assigning usually higher negative reward for each action which changes the active state s^a and effectively lowering the total given reward. The higher negative rewards of the deliberation cost represent the idea that changes of an active state are "harder to do" and that "thinking costs resources"* [10] ***ALTERNATIVE*** *Values of R consist of two parts $R(\tilde{s}, a, s) =$*

[9] Was the a^a a good idea?

[10] How to talk about this reward, should I introduce a new deliberation function, or otherwise it is not zero...

$R'(\tilde{s}, a, s) - D(\tilde{s}, a, s)$, where the R' function is a generalisation of the individual R_k reward functions of the $\mathcal{M}_k$ single systems, meaning that the R' is zero for non-active systems and the active system is rewarded on its own. The D function is then called the deliberation cost and it assigns a larger (in comparison to the reward)[11] values to the triplets $(\tilde{s}, a, s)$ where $s^a \neq a^a$. This effectively means that the actions, where the active single system is changed are taken as highly unprofitable, when only one time epoch is considered. This represents the idea that the changes of the active system are "harder to do" or that "thinking costs resources". [12][13]

This class of MDPs is defined as to be consistent with an idea that we are faced with a decision-making problem on a several *single system* MDPs among which we can switch at the beginning of each time epoch. The active *single system* $\mathcal{M}_k$, $k \in \mathbf{K}$ that is to be interacted with is chosen via the a^a-part of the action and the $\mathcal{M}_k$ then behaves as described in Section 4.1 with its corresponding sets $\mathbf{T_{a^a}}, \mathbf{S_{a^a}}, \mathbf{A_{a^a}}$ and functions P_{a^a}, R_{a^a}, which are determined by the original *multi-armed bandit* sets $\mathbf{T}, \mathbf{S}, \mathbf{A}$ and functions P, R.

It is worth noting that in this general case each *single system* MDP constructing the *multi-armed bandit* MDP can have different dimensions of the state and action space $\mathbf{S_i}$, $\mathbf{A_i}$, $i \in \{1, 2, ..., l\}$.

Example 2. *A real life example of this MDP could be again a biased coin tossing. Imagine a game when you are sitting at the table on which there are multiple coins with generally different bias. You get one coin on certain side in your hands and you are told the rules. You get \$1 each time the coin lands on Heads. You can choose to change your coin for any other coin for \$3 and you have to place it to the side it is on the table (you have to "conserve its state"). The turn of the coin costs 5 cents and the result of the toss generally depends on the side from which the coin is tossed. You know the bias of each one coin and the question is: what is the best sequence of actions that you can perform to get as much money as possible, on average, from this N-round game?*

It is important to mention that you are able to see the other coins "on the table" so that the position state s^p is known in each time epoch.

This Example 2 could be easily generalised to a process where some coins are dices or other $|\mathbf{S_k}|$-dimensional, $k \in \{1, 2, ..., |\mathbf{K}|\}$, entities expressing randomness in readers mind.

Now, when the optimality of a decision making on the *multi-armed bandit* MDPs is studied, a strong mathematical tool can be introduced. When there is a solution to a one class of problems, in this case decision making for *single system* MDPs, and another class is to be studied, one can try to interpret the latter problem as the former one and derive the solution with ease. This idea leads to interpretation of the *multi-armed bandit* MDP as *single system* through an easily imagined bijective mapping. [14]

When the *multi-armed bandit* MDP is interpreted as *single system* MDP the state and action space that describes the process is generally large. In fact so large that even standard dynamic programming is infeasible when the presented sparse MDP structure is not exploited. The exploitation of the *multi-armed bandit* MDP structure promises development of feasible algorithm for known-parameter case and thus also for the CE version for unknown-parameter case. All of these are unfortunately out of scope of this work and they are left for the future research.

On the top of a standard optimisation of *multi-armed bandit* MDPs for known and unknown P several modifications could be presented. One possible modification is that the first dimension of the action a^a provides a new information about the system that can be used for performing a more profitable a^p action. Or a second modification, where we are not able to observe the whole positional state s^p, but only the

[11] Actually other prices, but I have not introduced them.

[12] Please review this part and tell me what you thin about it...

[13] They can however be profitable in long-time horizon.

[14] Is it easy?

position of the active *single system*. Both of these are also out of scope of this work and they are left for the future research.

Chapter 5

Experiments

In this chapter the simulations of the MDPs that are studied in the previous chapter are presented. The purpose of these simulations is to illustrate the use of the presented theory in the previous chapter and to test the experimentally derived sub-optimal policies.

In both following sub-chapters there are simulations of *single system* MDPs. At first, I present two simulations of the *single system* with known transition probability function P, where the quality of the experimentally derived optimal policy by Algorithm 1 is the only studied feature.

The second part of this chapter focuses on simulation of a *single system* MDP, where the transition probability function P is not known. In the previous chapter two feasible approaches to the approximation of P and to the derivation of the sub-optimal strategy are presented. The certainty equivalence approach presented in Section 4.2.3 and the m-Step approximation in Section 4.2.4. The profitability of these sub-optimal policies derived by the respective methods is compared, while the optimal average cumulative reward is also presented.

It needs to be said that the simulation of the most complex optimisation problem, for the *multi-armed bandit* with unknown parameters, is out of the scope of this work. [1]

Now, the common ground of all the following experiments is presented. At first the actual experiment has to be defined. The type of MDP, the policy design and the possible type of estimation determines the program that is used for the experiment. All the programs that are used for the evaluation are entirely my work and they are attached to this thesis on a CD.

When the class of the simulation is chosen, a specific MDP $\mathcal{M}$, Definition 3.1, is defined through the supporting functions based on the dimensions of $\mathcal{M}$. Beside the definition of all the sets constructing the MDP, $\mathcal{M} = (\mathbf{T}, \mathbf{S}, \mathbf{A}, P, R)$ and an initial state $s_0 \in \mathbf{S}$, the prior information $w \in \mathbf{W}$, (for simulations with unknown P) has to be given by the statistician (or by a random generator).

The optimisation problem that is studied is thus well defined and the actual simulation of the decision making takes place. The main simulations that are run are focused on the derivation of the average cumulative rewards for the respective policy designs. For this purpose, the input-output loop of decision making, presented in Fig. **??** is embedded into a Monte-Carlo loop as we can see in Fig. **??**.

All the parts of the loop situated under the "Decision Maker" rectangle have already been defined by the choice of the experiment. These three parts of the decision making determine the sub-optimal action in each time epoch with respect to the up-to-date[2] knowledge about the simulated system (data, prior information) via respective algorithms.

After the sub-optimal action in each time epoch is taken, the probability distribution of the pseudo-random generator is set to the values corresponding with that action. The output is then given based

[1] It still is right? But now the reason is the complexity...

[2] Not sure about this...

on this distribution and it is called an "Observed state". The information about the input-action-output triplet $(s, a, \tilde{s})$ can then be treated in two ways. Either it is only assigned a reward with an R function, which is then used to make conclusion about the quality of the performed policy, or it can be also used for updating the sufficient statistic [3] v_t used for the evaluation of the P_t predictors, as can be seen in Equation (4.4).[4]

The evaluation of the experiments is mostly visual. The quality of policies tested in all simulations is rated based on histograms of frequencies of cumulative rewards, which are compared to the achievable optimum. The reason for using histograms is that the overall cumulative reward is a random variable and at least several hundreds iterations of the same simulation have to be run in order to get a reasonable statistic about the mean value. This Monte Carlo methodology, [**?**], is a standard way to simulation-based verification and comparison of alternative policies.

Also for the simulations with unknown P, the convergence of the P_t predictors, Fig. **??**, is presented by scatter graph as an evolution in time for the respective P_t values for one run of the simulation.

5.1 Optimal policy for a single system with known parameters

Experiment In this section two Monte Carlo simulations of a *single system* MDP with known parameters, which differ in the number of iterations, are presented.[5] As said in the beginning of this chapter the goal of this simulation is to check the quality of the derived sub-optimal policy. The overall structure of the experiment is described by the Fig. **??** and the detailed procedure for derivation of the sub-optimal actions are described in Algorithm 1. Because the class of the MDP that is studied is already determined, we can proceed to the definition of the dimensions, which are chosen as follows.

Settings and results The number of time epochs is chosen as $N = 50$ and the number of possible states and actions $|\mathbf{S}| = |\mathbf{A}| = 6$. This means that the functions R and P are defined by $|\mathbf{A}||\mathbf{S}|^2 = 216$ values. Because a table with 216 values would be of a little use, these values are presented only in a digital form in the file *DynamicKnown.mat* on a CD attached to this thesis.

The first number of iterations for the Monte Carlo method is set to 500, while the second run is performed for 5 000 simulations of the process. [6]

The quality of the derived sub-optimal policy can be visually checked with both histograms of frequencies of the cumulative rewards obtained from both Monte Carlo simulations. These histograms can be seen in Fig. **??** and Fig. **??**.

The optimal average cumulative reward determined by the optimal value function is 85.2003, whereas the average values of my simulations are 85.6264 for the run with 500 Monte Carlo iterations, and 85.2236 for the run with 5 000 iterations. [7]

Discussion The main goal of this simulation is to check the quality of the experimentally derived sub-optimal policy via Algorithm 1. As we can see, the difference between the average cumulative reward of the derived policy and the expected optimal average cumulative reward is approximately 0.5% in the first simulation with 500 Monte Carlo iterations and approximately 0.026% in the second simulation with 5000 iterations. [8]

[3] Information state?

[4] Or in algorithms 3 and 4?

[5] Is it called like that? Iterations?

[6] Or the N-round process?

[7] Maybe to much digits?

[8] I dont know how to fit the exceedance in here, so if it is not that important I would prefer not to talk about it.

We can also see that in the second performed experiment made for 5 000 iterations, ten times the first case, both the variance of the distribution and the difference from the optimal value decreased. It is then assumed that this trend would continue with more iterations and that both the variance and the difference would get closer and closer to zero.

With this assumption we can say that the experimentally derived optimal policy is in fact the optimal one. [9]

5.2 Optimal policy for a single system with unknown parameters

In this section a second simulation of the decision making is presented. Now the *single system* MDP with unknown transition probability function P is simulated. It is important to mention that the lack of information about P is just pretended. If we want to compare our experimentally derived decision policies to the optimal one, we have to know what the optimum is.

In the previous chapter we have derived two different approaches to the approximation of the unknown P function. The first one is the CE approach, Section 4.2.3, and the second one is the m-Step approach, Section 4.2.4.

The former is simulated in the first sub-section, where the average experimental cumulative reward is compared to the achievable optimum derived by Algorithm 1, for known P. In addition the first sub-chapter offers a grid of graphs that show the evolution of the predictors P_t over time. This can be seen in Fig. **??**.

The m-Step approach to the P estimation is then simulated in the second sub-chapter, where the average experimental cumulative rewards are again compared to the achievable optimum. On the top of that the results of m-Step approximation are also compared with the ones of CE approach. The convergence of the predictor is no longer studied in this sub-chapter, as the results would be merely the same as in the CE case.

5.2.1 Algorithm based on certainty equivalence

Experiment In this sub-section the simulation of a decision making on a *single system* MDP $\mathcal{M}$ with unknown parameters, when the CE approach to the P estimation is used is presented. The first experiment of this section focuses only on one run of the defined MDP and provides a grid of graphs, where the convergence of P_t predictor values is shown. The second experiment of this section focuses on the quality of the sub-optimal policy derived by Algorithm 3. The comparison to the achievable optimum is depicted in Fig. **??**.

Settings and results The dimensions of the specific MDP that is studied are for both parts of the experiment the same. The cardinality of sets **A** and **S** is set to 2, so that the convergence of the predictors P_t to P can be reasonably visualised, see Fig. **??**. This means that both P and R are represented by 8 values which can both be found in Table 5.1.

[9] Is that a good reasoning?

$P(:,:,s=1)$		
$\tilde{s}/a$	1	2
1	0.4237	0.6885
2	0.5763	0.3115

$R(:,:,s=1)$		
$\tilde{s}/a$	1	2
1	1.4315	-0.0569
2	2.7287	-0.3111

$w(:,:,s=1)$		
$\tilde{s}/a$	1	2
1	8	5
2	1	2

$P(:,:,s=2)$		
$\tilde{s}/a$	1	2
1	0.7282	0.8816
2	0.2718	0.1184

$R(:,:,s=2)$		
$\tilde{s}/a$	1	2
1	-0.1039	2.5921
2	1.1251	-0.2876

$w(:,:,s=2)$		
$\tilde{s}/a$	1	2
1	1	2
2	9	1

Table 5.1: Tables of P, R and w values for the CE approach simulation. Original 3D arrays are presented by two "slices" which differ in the initial state $s = 1$ or $s = 2$.

[10]

The number of time epochs N is set to $N = 50$, so that the convergence of P_t predictor values could easily be seen. The number of iterations for deriving the quality of the experimentally obtained sub-optimal policy by Monte Carlo method is set to 1500 in the second experiment of this section.

The initial state is chosen as $s_0 = 2$ and the parameter values $w(\tilde{s}, a, s)$ that determine the prior distribution can also be found in Table 5.1.

The distribution of the cumulative reward for the CE derived policy can be seen in Fig. **??** and the convergence of P_t to P can be seen in Fig. **??**.

The average cumulative reward for the CE approach to this MDP $\mathcal{M}$ given by the experiment is 110.18, whereas the optimal value is 110.40 .

Discussion In the simulations in this section, both the convergence of predictors P_t to P, and the quality of the derived sub-optimal policy for the *single system* with unknown P is shown. In Fig. **??** we can see that the convergence of some predictor values to the real P values is fast for some values, in our case for pairs $(s = 1, a = 1)$ and $(s = 2, a = 2)$, and sometimes it is non-existent. This is due to the strategy that is performed through the experiment. The algorithm evaluates the best sub-optimal actions from its knowledge at the time of a decision. This results, in this case, to taking the action $a = 2$ every time the process is in state $s = 2$, and also action $a = 1$ when the state is $s = 1$.

Due to this decision-making strategy the values of the triplets $(\tilde{s}, a, s)$ do or do not change based only on the fact if the actions that form them are performed or not.

In the second simulation we can see, that the optimal average cumulative reward 110.40 is not far from the experimentally obtained one 110.18. The reason for [11] this is that the prior estimates are not strong and the number of time epoch $N = 50$ is high enough for the algorithm to "learn" the P correctly.[12]

[10] Hope this representation is OK, and how to do the backslash $\tilde{s}, a$?

[11] of?

[12] Maybe other word than strong

This means that evaluated fractions for the predictor values, as described by Equation (4.4), are highly adaptive to the actually measured triplets $(\tilde{s}, a, s)$. This strong adaptivity results in a high-quality semi-optimal strategy.

5.2.2 Algorithm based on m-Step improvement of certainty equivalence

Experiment The last experiment that is presented is the simulation of the decision making on the *single system* MDP $\mathcal{M}$, where the m-Step approximation approach for P estimation is compared to the CE one and to the optimum. New MDP is defined and the quality of the respective approaches is again determined by the Monte Carlo method.

Settings and results The number of time epochs is chosen as $N = 30$ and the number of possible states and actions $|\mathbf{S}| = |\mathbf{A}| = 3$. This means that the functions R and P are defined by $|\mathbf{A}||\mathbf{S}|^2 = 27$ values. Because a table with 27 values would still be of a little use, these values are presented only in a digital form in the file *MCCEvsmStep.mat* on a CD attached to this thesis.

The number m that is crucial for this method is set to $m = 3$ as it is the maximal number for a feasible computation (in order of hours on my computer). The number of Monte Carlo iterations is set to 1000.

The comparison of the average cumulative rewards for the CE estimation approach and the m-Step estimation approach can be seen in histogram Fig. **??**, where also the optimal value is depicted.
[13]

The value of m-Step approximation average cumulative reward determined by my simulation is 39.60, the value of CE approximation average is 38.69 and the optimal value is 54.63.

Discussion The main reason for construction of this experiment is to confirm the superiority of the m-Step approximation approach to the CE one. As we can see above, the average cumulative reward for m-Step approach is up to 2.4% higher than the one derived by the CE approximation. This number is derived from the 1000 iterations Monte Carlo experiment. The superiority of the m-Step approximation can be also seen visually in histogram Fig. **??**, where the frequencies of the m-Step approach are higher with the higher values of the cumulative rewards obtained in one run.

Also, when compared to the maximum, our experimentally derived average cumulative rewards from each estimation are approximately 70% of the optimum. I do think that this is due to the higher number of possible paths of the MDP presented in this section in comparison with the previous one. A higher number of possible actions means higher number of the non-optimal ones. In the case, where there are two actions to choose from one is optimal and the other is not. In a MDP where $|\mathbf{A}| = 3$ the number of non-optimal actions doubles in comparison to MDP with $|\mathbf{A}| = 2$. [14]

The question whether the 2.4% improvement is significant depends of the frame of reference. If this algorithm would be theoretically used in some part of a billion dollar industry, where the 2.4% improvement would mean tens of millions of dollars, the difference would indeed be significant. However in a environment, where the revenue from the decision making would be low, and for example the computing power is expensive, the implementation of m-Step algorithm is maybe not worth the effort and the much simpler CE approach for P_t predictor estimation is sufficient. This is actually also a decision-making problem, what approach to use, when the price of computing power is given together with the expected revenue from the different estimation approaches.

[13] It was red in the eps. file, now it is black, What to do with it?

[14] I could talk about the estimates, which were like the lowest possible, but it would not fit my reasoning above.

Chapter 6

Discussion

In this chapter an overview of the performed experiments from the Chapter 5 is presented.

The purpose of each experiment is to check the quality of the policies derived theoretically in Chapter 4, via the average cumulative rewards, obtained by the Monte Carlo method.

At first, the quality of the policy derived for the *single system* with known P, is tested. The average cumulative reward is obtained for two numbers of Monte Carlo iterations [1], Section 5.1, where each time the difference between the optimal average cumulative reward and the experimental one is lower than 0.5%. The reason why two experiments with different number of Monte Carlo iterations are run is that we are able to see a trend of [2] the histograms. With higher number of iterations the histogram becomes more narrow and the difference of the averages decreases. Since the difference is as low as 0.5% and it is getting lower with more iterations, we can say that the derived policy via algorithm 1 is really the optimal one. [3]

The second part of the experiments is devoted to the study of two different approaches for obtaining the optimal decision policy for *single system* with unknown P. These two approaches, the CE approximation, Section 4.2.3, and the m-Step approximation, Section 4.2.4, are tested again in a sense of their quality, with a hope, that the theoretically superior m-Step approximation approach will produce higher average cumulative reward than the CE approximation.

This hypothesis was confirmed, as we can see in Section 5.2.2, histogram Fig. **??**. The m-Step approximation gave up to 2.4% larger average cumulative reward than the CE approach, when sustaining a reasonable demand for computing power. This would of course change if the settings of MDPs in experiment Section 5.2.2 had such extreme dimensions that even 1-step approximation would not be feasible.

Also an experiment, which goal is to visualise the evolution of the predictor P_t is performed in Section 5.2.1. In the grid of graphs Fig. **??**, we can see, how some values of the predictor converge with time, while some values of the P_t predictors stay unchanged through the whole process.

The problem of the non-adaptivity of certain values of the predictors is that the paths that determine their value are never tried, even though they could possibly be the most profitable. This problem is called the exploration vs. exploitation problem and its solution is the optimal rate between trying new paths and learning the true nature of the process, and the exploitation of the sub-most profitable (according to

[1] Not sure about this formulation

[2] in?

[3] Is that a good reasoning?

our limited knowledge) paths. [4] The study of this dichotomy [5] is out of scope of this work, and it is left for the future research.

[4] Well I have it in abstract and this is the first time i have talked about it, is that a problem?
[5] really dichotomy?

Chapter 7

Conclusions

[1] In this Bachelor's thesis [2] I study the optimal decision making on discrete Markov decision processes (MDPs). I have defined two classes of MDPs which I call *single system* and *multi-armed bandit*, which differ by the structure of their respective sets.

I have presented the theory of dynamic programming as a key method for finding the optimal policies [3] for *single system* MDPs with known transition probability function P. This theory is then confirmed by an experiment in Section 5.1, where the quality of the derived sub-optimal policy is tested. The difference between the experimentally derived average cumulative reward, obtained by Monte Carlo method, and the optimal average cumulative reward, determined by the value function, is lower than 0.5% for 500 Monte Carlo iterations. This difference even decreased to 0.026% for 5000 Monte Carlo iterations. This decreasing, almost negligible, difference serves as an argument for the verification of the optimality of the derived policy.

The second part of the thesis focused on decision making on the *single system* MDPs, when the transition probability function P is not known. Three solutions to this decision-making problem are presented. In all of them the estimation of the unknown P in a form of predictors P_t, Section 3.3, is based on the Bayesian theory for estimation of the parameter θ with which the P is parametrized. The difference in the three solutions for this case of unknown P is in both computational complexity and the quality of the derived policies.

The first approach called optimal Bayesian approach, Section 4.2.2, is based on the construction the generalised state (s, v_t), where this v_t is called the information state and which represents the history of the process until the time t. Based on this information state v_t and prior information for the Bayesian estimation w we are able to construct predictors needed for the dynamic programming for each time epoch for all the possible paths of the process. I have also argued, that this approach is infeasible one, because of the exponential increase of the complexity of dynamic programming due to the increase in the cardinality of the informational state v_t with time. [4]

The second approach to the estimation that I have presented is called the certainty equivalence (CE) approach. The main purpose of this approach is to lower the complexity of the algorithm that derives the optimal policy in each time epoch of the process. The predictors that are used for the dynamic programming are presented as "frozen in time", meaning that the predictor that is used for the evaluation of the value function in the last time epoch is the same as the one for the first one. This approach

[1] As I had time on Monday and I think you will not like this conclusions I offer a second one right after this one.
[2] Should I change it to bachelors degree project?
[3] Should there be references here?
[4] references??

significantly decreases the computational power needed for the computation of the, now sub-optimal, policy.

As this CE approximation is rather rough [5] we can expect that the derived sub-optimal policy will not provide as good average cumulative rewards as the optimal one. The quality of the sub-optimal policy derived by the CE approximation method, Algorithm 3, is performed in Section 5.2.1, and in spite of the expectation the average cumulative reward of the experimentally derived sub-optimal policy is lower than the optimal value only by 0.2%. In Section 5.2.1 I have argued that the reason of this high quality of the derived policy is due to the high adaptivity of the algorithm, which is determined by the use of weak prior estimation in my simulation. The convergence of the predictors P_t, Fig. **??** is rather fast and thus the decisions of the optimal policy are performed by the derived sub-optimal one from the early time epochs of the experiment.

The third approach, that I have come up with, for the decision making on the *single system* MDP that I present in this thesis is called the m-Step approximation. The purpose of this approach is to exploit the computational power available for even better results that the CE approximation gives. The idea behind this m-Step approach is that the predictors P_t are not "frozen" completely, but only after limited amount of time epochs m, in which they are expected to be changing. This approach is a hybrid between the first and second approach and it allows the statistician to maximise the quality of the derived sub-optimal policy with respect to the computing power they possess.

A simulation for the *single system* processes, where the quality of second, CE, and third, m-Step, approach is compared to the optimum is presented in Section 5.2.2. It is shown that for a higher dimensional process (in this case $|\mathbf{S}| = |\mathbf{A}| = 3$) the difference between the CE and m-Step approach is relatively large, up to 2.4% for the studied *single system* MDP. This simulation, only with $m = 3$, confirmed the superiority of the m-Step algorithm to the CE one, while the computational power of my computer is enough to perform the computation.

Second class of MDPs, called the *multi-armed bandit* MDP is coped with only theoretically in Section 4.3. It is shown that in terms of mathematical computation, the optimal and sub-optimal policies for this class of the MDPs can be derived in the same way that the *single system* class MDPs are. The reason why this class is introduced, is the structure of its members. The structure of the *multi-armed bandit* MDP better describes the structure of a decision making on more than one entity and the theory of this type of MDPs can be widened as is described in the end of the Section 4.3.

[6]

SECOND CONCLUSIONS

In this bachelor's thesis I have presented a unified theory for decision making on the discrete Markov decision processes (MDPs). I have defined two classes od the MDPs which I call *single system* MDPs, Definition **??**, and *multi-armed bandit* MDP. The theory of MDPs and its notation is inspired by Putterman's book on this topic, [9].

The main focus of this thesis is to derive the optimal policy, Definition **??**, for the defined classes of MDPs. Two cases of the optimisation problems are presented, where in both the theory of dynamic programming, taken from Bellman's book [**?**], is used.

In the first part of the thesis the Bellman's theory is applied to the *single system* MDPs where the transition probability function P is expected to be known, Section **??**. The quality of the derived policy is tested in Section **??**, where the negligible difference between the experimentally derived average cu-

[5] Hruby

[6] Should I talk more about the future research and if so, what should it be? Beside the dependency in the multi-armed bandit. What else can be done. My solution is not great, the improvement of the mStep algorithm must be forced by certain specific values, so that the basic CE is the best because it is easy and it does not make worse results than the mStep much. Both are not bad in comparison with the opitmum for longer runs and for not-strong estimates.

mulative reward served as an argument of declaring the experimentally derived sub-optimal policy as the optimal one.

In the second part of this thesis I present a problem of optimisation for *single systems* with unknown transition probability function P, Section **??**, to which I present 3 different solutions. In all solutions the Bayesian estimation theory for P is used. The difference in the solutions is the use of the Bayesian estimation in the dynamic programming algorithm, while the details are explained below.

The first solution that I present for the *single system* optimal decision-making problem is called optimal Bayesian approach, Section **??**. This solution is based on the introduction of the informational state v_t which serves as a statistic describing the history of the MDP. A generalised notion of "state" is defined, **??**, and the dynamic programming algorithm is used in the same way as for the case with known P. It is shown that this approach works in theory, but it extremely increases the need for a computational power. Thus a second approach is presented.

The second approach that I present... etc. **I will make the review of your review as suggested, but the idea of the second try is clear I suppose**

Bibliography

[1] Michael O'Gordon Duff. *Optimal Learning; Computational Procedures for Bayes-Adaptive Markov Decision Processes*. PhD thesis, 2002.

[2] Leonardo Azevedo Fernando Luis Bordignon, Leandro Passos de Figueireido. Hybrid global stochastic and Bayesian linearized acoustic seismic inversion methodology. *IEEE Transactions on Geoscience and Remote Sensing*, 55:4457–4464, August 2017.

[3] Zvi Galil and Raffaele Giancarlo. Speeding up dynamic programming with applications to molecular biology. 1987. Elsevier.

[4] Jaromir Simsa Jiri Herman, Radan Kucera. *Counting and Configurations: Problems in Combinatorics, Arithmetic, and Geometry*. Springer Science & Business Media, January 2003.

[5] Mohri Mehryar Joannès Vermorel. Multi-armed bandit algorithms and empirical evaluation. In *Machine Learning: ECML 2005*.

[6] Edmund Landau. *Handbuch der Lehre von der Verteilung der Primzahlen*, volume 01. Liepzig B.G. Teubner, 1909.

[7] Nouha Jaoua Marwa Chaabane, Majdi Mansouri. Bayesian method for states estimation in structural health monitoring application. Monastir, Tunisia, 2016. IEEE.

[8] V. Peterka. Bayesian approach to system identification. In *in Trends and Progress in System Identification, P. Eykhoff, Ed*, pages 239–304. Pergamon Press, 1981.

[9] M.L. Puterman. *Markov Decsion Processes: Discrete Stochastic Dynamic Programming*. Wiley, 2005.

[10] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, 1989.

[11] Marko Ruman. Decision making in multi-proposer ultimatum game, 2016. CTU FJFI.

[12] Sidney Yakowitz. Dynamic programming applications in water resources. *AN AGU JOURNAL*, 18, 1982. draft.