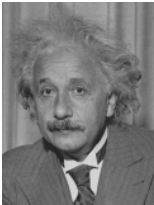


I have no particular talent. I am merely inquisitive.
— Albert Einstein



Albert Einstein
(1879–1955)

1 A Short Tour Through the Book

1.1	Introduction	2
1.2	Simplicity & Uncertainty	3
1.2.1	Introduction	3
1.2.2	Algorithmic Information Theory	4
1.2.3	Uncertainty & Probabilities	5
1.2.4	Algorithmic Probability & Universal Induction	6
1.2.5	Generalized Universal (Semi)Measures	7
1.3	Universal Sequence Prediction	7
1.3.1	Setup & Convergence	8
1.3.2	Loss Bounds	8
1.3.3	Optimality Properties	9
1.3.4	Miscellaneous	10
1.4	Rational Agents in Known Probabilistic Environments.....	11
1.4.1	The Agent Model	11
1.4.2	Value Functions & Optimal Policies	11
1.4.3	Sequential Decision Theory & Reinforcement Learning..	12
1.5	The Universal Algorithmic Agent AIXI	13
1.5.1	The Universal AIXI Model	13
1.5.2	On the Optimality of AIXI.....	14
1.5.3	Value-Related Optimality Results	15
1.5.4	Markov Decision Processes	17
1.5.5	The Choice of the Horizon	18
...		

1.6	Important Environmental Classes	18
1.6.1	Introduction	18
1.6.2	Sequence Prediction (SP)	19
1.6.3	Strategic Games (SG)	19
1.6.4	Function Minimization (FM)	19
1.6.5	Supervised Learning from Examples (EX)	19
1.6.6	Other Aspects of Intelligence	20
1.7	Computational Aspects	20
1.7.1	The Fastest & Shortest Algorithm for All Problems	20
1.7.2	Time-Bounded AIXI Model	22
1.8	Discussion	24
1.9	History & References	26

This Chapter represents a short tour through the book. It is not meant as a gentle introduction for novices, but as a condensed presentation of the most important concepts and results of the book. The price for this brevity is that in this chapter we mostly forgo mathematical rigor, subtleties, proofs, discussions, references and comparisons to other work. More seriously, some sections demand high background knowledge. Readers unfamiliar with algorithmic information theory should first read Chapter 2 or consult the textbooks [LV97, Cal02]. Readers unfamiliar with sequential decision theory should first read Chapter 4 or consult the textbooks [BT96, SB98]. Before becoming discouraged by the complexity of some of the sections, it is better to skip them completely.

1.1 Introduction

Artificial Intelligence. The science of artificial intelligence (AI) might be defined as the construction of intelligent systems and their analysis. A natural definition of a *system* is anything that has an input and an output stream. Intelligence is more complicated. It can have many faces like creativity, solving problems, pattern recognition, classification, learning, induction, deduction, building analogies, optimization, surviving in an environment, language processing, knowledge and many more. A formal definition incorporating every aspect of intelligence, however, seems difficult. Further, intelligence is graded: There is a smooth transition between systems, which everyone would agree to be not intelligent, and truly intelligent systems. One simply has to look in nature, starting with, for instance, inanimate crystals, then amino acids, then some RNA fragments, then viruses, bacteria, plants, animals, apes, followed by the truly intelligent homo sapiens, and possibly continued by AI systems or ETs. So, the best we can expect to find is a partial or total order relation on the set of systems, which orders them w.r.t. their degree of intelligence (like

intelligence tests do for human systems, but for a limited class of problems). Having this order we are, of course, interested in large elements, i.e. highly intelligent systems. If a largest element exists, it would correspond to the most intelligent system which could exist.

Most, if not all, known facets of intelligence can be formulated as goal driven or, more precisely, as maximizing some utility function. It is therefore sufficient to study goal-driven AI. For example, the (biological) goal of animals and humans is to survive and spread. The goal of AI systems should be to be useful to humans. The problem is that, except for special cases, we know neither the utility function nor the environment in which the agent will operate in advance.

Main idea. This book presents a theory that formally¹ solves the problem of unknown goal and environment. It might be viewed as a unification of the ideas of universal induction, probabilistic planning and reinforcement learning, or as a unification of sequential decision theory with algorithmic information theory. We apply this model to some of the facets of intelligence, including induction, game playing, optimization, reinforcement and supervised learning, and show how it solves these problem classes. This, together with general convergence theorems, supports the belief that the constructed universal AI system is the best one in a sense to be clarified in the following, i.e. that it is the most intelligent environment-independent system possible. The intention of this book is to introduce the universal AI model and give an extensive analysis.

1.2 Simplicity & Uncertainty

This section introduces Occam's razor principle, Kolmogorov complexity, and objective/subjective probabilities. We finally arrive at the problem of universal prediction, and its solution by Solomonoff.

1.2.1 Introduction

An important and nontrivial aspect of intelligence is inductive inference. Simply speaking, induction is the process of predicting the future from the past, or, more precisely, it is the process of finding rules in (past) data and using these rules to guess future data. Weather or stock-market forecasting or continuing number series in an IQ test are nontrivial examples. Making good predictions plays a central role in natural and artificial intelligence in general, and in machine learning in particular. All induction problems can be phrased

¹ With a formal solution we mean a rigorous mathematical definition, uniquely specifying the solution. In the following, a solution is always meant in this formal sense.

as sequence prediction tasks. This is, for instance, obvious for time-series prediction, but also includes classification tasks. Having observed data x_t at times $t < n$, the task is to predict the n^{th} symbol x_n from sequence $x_1 \dots x_{n-1}$. This *prequential approach* [Daw84] skips over the intermediate step of learning a model based on observed data $x_1 \dots x_{n-1}$ and then using this model to predict x_n . The prequential approach avoids problems of model consistency, how to separate noise from useful data, and many other issues. The goal is to make “good” predictions, where the prediction quality is usually measured by a loss function, which shall be minimized. The key concept to well-defining and solving induction problems is *Occam’s razor* (simplicity) principle, which says that “*Entities should not be multiplied beyond necessity.*” This may be interpreted as keeping the simplest theory consistent with the observations $x_1 \dots x_{n-1}$ and using this theory to predict x_n . Before we can present Solomonoff’s formal solution, we have to quantify Occam’s razor in terms of Kolmogorov complexity, and introduce the notions of subjective and objective probabilities.

1.2.2 Algorithmic Information Theory

Intuitively, a string is simple if it can be described in a few words, like “the string of one million ones”, and is complex if there is no such short description, like for a random string whose shortest description is specifying it bit by bit. We can restrict the discussion to binary strings, since for other (non-stringy mathematical) objects we may assume some default coding as binary strings. Furthermore, we are only interested in effective descriptions, and hence restrict decoders to be Turing machines. Let us choose some universal (so-called prefix) *Turing machine* U with unidirectional binary input and output tapes and a bidirectional work tape. We can then define the *prefix Kolmogorov complexity* [Cha75, Gács74, Kol65, Lev74] of a binary string x as the length ℓ of the shortest program p for which U outputs the binary string x

$$K(x) := \min_p \{\ell(p) : U(p) = x\}.$$

Simple strings like 000...0 can be generated by short programs, and, hence have low Kolmogorov complexity, but irregular (e.g. random) strings are their own shortest description, and hence have high Kolmogorov complexity. An important property of K is that it is nearly independent of the choice of U . Furthermore, it shares many properties with Shannon’s entropy (information measure) S , but K is superior to S in many respects. Figure 2.11 on page 38 contains a schematic graph of K . To be brief, K is an excellent universal complexity measure, suitable for quantifying Occam’s razor. There is (only) one severe disadvantage: K is not finitely computable. More precisely, a function f is said to be *finitely computable* (or *recursive*) if there exists a Turing machine which, given x , computes $f(x)$ and then halts. Some functions are not finitely computable but still *approximable* in the sense that there is a nonhalting Turing machine with an infinite output sequence y_1, y_2, y_3, \dots with $\lim_{t \rightarrow \infty} y_t = f(x)$.

If additionally the output sequence is monotone increasing/decreasing, then f is said to be *lower/upper semicomputable* (or *enumerable/co-enumerable*). Finally, we call f *estimable* if some Turing machine, given x and a precision ε , finitely computes an ε -approximation of x . The major algorithmic property of K is that it is co-enumerable, but not finitely computable.

1.2.3 Uncertainty & Probabilities

For the *objectivist*, probabilities are real aspects of the world.² The outcome of an observation or an experiment is not deterministic, but involves physical random processes. Kolmogorov's axioms of probability theory formalize the properties which probabilities should have. In the case of independent and identically distributed (i.i.d.) experiments the probabilities assigned to events can be interpreted as limiting frequencies (*frequentist* view), but applications are not limited to this case. Conditionalizing probabilities and Bayes' rule are the major tools in computing posterior probabilities from prior ones. For instance, given the initial binary sequence $x_1 \dots x_{n-1}$, what is the probability of the next bit being 1? The probability of observing x_n at time n , given past observations $x_1 \dots x_{n-1}$ can be computed with multiplication or the chain rule³ if the true generating distribution μ of the sequences $x_1 x_2 x_3 \dots$ is known: $\mu(x_n | x_{<n}) = \mu(x_{1:n}) / \mu(x_{<n})$, where we introduced the abbreviations $x_{1:n} \equiv x_1 x_2 \dots x_n$ and $x_{<n} \equiv x_1 x_2 \dots x_{n-1}$. The problem, however, is that one often does not know the true distribution μ (e.g. in the cases of weather and stock-market forecasting).

The *subjectivist* uses probabilities to characterize an agent's degree of belief in (or plausibility of) something, rather than to characterize physical random processes. This is the most relevant interpretation of probabilities in AI. It is somewhat surprising that plausibilities can be shown to also respect Kolmogorov's axioms of probability and the chain rule by assuming only a few plausible qualitative rules they should follow [Cox46]. Hence, if the plausibility of $x_{1:n}$ is $\rho(x_{1:n})$, the degree of belief in x_n given $x_{<n}$ is, again, given by the chain rule: $\rho(x_n | x_{<n}) = \rho(x_{1:n}) / \rho(x_{<n})$.

The chain rule allows the computation of posterior probabilities/plausibilities from prior ones, but leaves open the question of how to determine the priors themselves. In statistical physics, the principle of indifference (symmetry principle) and the maximum entropy principle can often be exploited to determine prior probabilities, but only Occam's razor is general enough to assign prior probabilities in *every* situation, especially to cope with complex domains typical for AI.

² Readers not believing in objective and/or subjective probabilities should read the remark at the beginning of Section 2.3.

³ Strictly speaking, it is just the definition of conditional probabilities.

1.2.4 Algorithmic Probability & Universal Induction

Occam's razor (appropriately interpreted and in compromise with Epicurus' principle of indifference) tells us to assign high/low a priori plausibility to simple/complex strings x . Using K as complexity measure, any monotone decreasing function of K , e.g. $\rho(x) = 2^{-K(x)}$, would satisfy this criterion. But ρ also has to satisfy the probability axioms, so we have to be a bit more careful. Solomonoff [Sol64, Sol78] defined the *universal prior* $M(x)$ as the probability that the output of a universal Turing machine U starts with x when provided with fair coin flips on the input tape. Formally, M can be defined as

$$M(x) := \sum_{p : U(p)=x^*} 2^{-\ell(p)} \quad (1.1)$$

where the sum is over all (so-called minimal) programs p for which U outputs a string starting with x . Strictly speaking M is only a *semimeasure* since it is not normalized to 1, but this is acceptable/correctable. We derive the following bound:

$$\sum_{t=1}^{\infty} (1 - M(x_t | x_{<t}))^2 \leq -\frac{1}{2} \sum_{t=1}^{\infty} \ln M(x_t | x_{<t}) = -\frac{1}{2} \ln M(x_{1:\infty}) \leq \frac{1}{2} \ln 2 \cdot Km(x_{1:\infty})$$

where $Km(x_{1:\infty})$ is the length of the shortest (nonhalting) program computing $x_{1:\infty}$. In the first inequality we have used $(1-a)^2 \leq -\frac{1}{2} \ln a$ for $0 \leq a \leq 1$. In the equality we exchanged the sum with the logarithm and eliminated the resulting product by the chain rule. In the last inequality we used $M(x) \geq 2^{-Km(x)}$, which follows from definition (1.1) by dropping all terms in \sum_p except for the shortest p computing x . If $x_{1:\infty}$ is a computable sequence, then $Km(x_{1:\infty})$ is finite, which implies $M(x_t | x_{<t}) \rightarrow 1$ ($\sum_{t=1}^{\infty} (1-a_t)^2 < \infty \Rightarrow a_t \rightarrow 1$). This means that if the environment is a computable sequence (whichever, e.g. the digits of π or e in binary representation), after having seen the first few digits, M correctly predicts the next digit with high probability, i.e. it recognizes the structure of the sequence.

Assume now that the true sequence is drawn from the distribution μ , i.e. the true (objective) probability of $x_{1:n}$ is $\mu(x_{1:n})$, but μ is unknown. How is the posterior (subjective) belief $M(x_n | x_{<n}) = M(x_n) / M(x_{<n})$ related to the true (objective) posterior probability $\mu(x_n | x_{<n})$? Solomonoff's [Sol78] central result is that the posterior (subjective) beliefs converge to the true (objective) posterior probabilities, if the latter are computable. More precisely, he showed that

$$\sum_{t=1}^{\infty} \sum_{x_{<t} \in \{0,1\}^{t-1}} \mu(x_{<t}) \left(M(0 | x_{<t}) - \mu(0 | x_{<t}) \right)^2 \leq \frac{1}{2} \ln 2 \cdot K(\mu) + O(1). \quad (1.2)$$

The complexity $K(\mu)$ is finite if μ is a computable function, but the infinite sum on the l.h.s. can only be finite if the difference $M(0 | x_{<t}) - \mu(0 | x_{<t})$ tends to zero for $t \rightarrow \infty$ with μ -probability 1 (w. μ .p.1). This shows that using M as an estimate for μ may be a reasonable thing to do.

1.2.5 Generalized Universal (Semi)Measures

One can derive a universal prior in a different way: Solomonoff [Sol64, Eq.(13)] defines a somewhat problematic mixture over all computable probability distributions. Levin [ZL70] considers the larger class $\mathcal{M}_U := \{\nu_1, \nu_2, \dots\}$ of all so-called enumerable semimeasures. Let $\mu \in \mathcal{M}_U$, and assign (consistent with Occam's razor) a prior plausibility of $2^{-K(\nu_a)}$ to ν_a . Then the prior plausibility of $x_{1:n}$ is, by elementary probability theory,

$$\xi_U(x_{1:n}) := \sum_{\nu \in \mathcal{M}_U} 2^{-K(\nu)} \nu(x_{1:n}). \quad (1.3)$$

One can show that ξ_U coincides with M within an (irrelevant) multiplicative constant, i.e. $M(x) \stackrel{\times}{\approx} \xi_U(x)$, where $f(x) \stackrel{\times}{\leq} g(x)$ abbreviates $f(x) = O(g(x))$, and $\stackrel{\times}{\approx}$ denotes $\stackrel{\times}{\leq}$ and $\stackrel{\times}{\geq}$. Both ξ_U and M can be shown to be lower semicomputable. The dominance $M(x) \stackrel{\times}{\approx} \xi_U(x) \geq 2^{-K(\mu)} \mu(x)$ is the central ingredient in the proof of (1.2). The advantage of ξ_U over M is that the definition immediately generalizes to arbitrary weighted sums of (semi)measures in \mathcal{M} for arbitrary countable \mathcal{M} . Most proofs in this book go through for generic \mathcal{M} and weights.

So, what is so special about the class of all enumerable semimeasures \mathcal{M}_U ? The larger we choose \mathcal{M} , the less restrictive is the assumption that \mathcal{M} should contain the true distribution μ , which will be essential throughout the book. Why not restrict to the still rather general class of estimable or finitely computable (semi)measures? For *every* countable class \mathcal{M} , the mixture $\xi(x) := \xi_{\mathcal{M}}(x) := \sum_{\nu \in \mathcal{M}} w_{\nu} \nu(x)$ with $w_{\nu} > 0$, the important dominance $\xi(x) \geq w_{\nu} \nu(x)$ is satisfied. The question is, what properties does ξ possess. The distinguishing property of \mathcal{M}_U is that ξ_U is itself an element of \mathcal{M}_U . On the other hand, in this book $\xi_{\mathcal{M}} \in \mathcal{M}$ is not by itself an important property. What matters is whether ξ is computable in one of the senses we defined above. There is an enumerable semimeasure (M) that dominates all enumerable semimeasures in \mathcal{M}_U . As we will see, there is *no* estimable semimeasure that dominates all computable measures, and there is *no* approximable semimeasure that dominates all approximable measures. From this it follows that for a universal (semi)measure which at least satisfies the weakest form of computability, namely being approximable, the largest dominated class among the classes considered in this book is the class of enumerable semimeasures, but there are even larger classes [Sch02a]. This is the reason why \mathcal{M}_U and M play a special role in this (and other) works. In practice though, one has to restrict to a finite subset of finitely computable environments ν to get a finitely computable ξ .

1.3 Universal Sequence Prediction

In the following we more closely investigate sequence prediction (SP) schemes based on Solomonoff's universal prior $M \stackrel{\times}{\approx} \xi_U$ and on more general Bayes

mixtures ξ , mainly from a decision-theoretic perspective. In particular, we show that they are optimal w.r.t. various optimality criteria.

1.3.1 Setup & Convergence

Let $\mathcal{M} := \{\nu_1, \nu_2, \dots\}$ be a countable set of candidate probability distributions on strings over the finite alphabet \mathcal{X} . We define a weighted average on \mathcal{M} :

$$\xi(x_{1:n}) := \sum_{\nu \in \mathcal{M}} w_\nu \cdot \nu(x_{1:n}), \quad \sum_{\nu \in \mathcal{M}} w_\nu = 1, \quad w_\nu > 0. \quad (1.4)$$

It is easy to see that ξ is a probability distribution as the weights w_ν are positive and normalized to 1 and the $\nu \in \mathcal{M}$ are probabilities. We call ξ universal relative to \mathcal{M} , as it multiplicatively dominates all distributions in \mathcal{M} in the sense that $\xi(x_{1:n}) \geq w_\nu \cdot \nu(x_{1:n})$ for all $\nu \in \mathcal{M}$. In the following, we assume that \mathcal{M} is known and contains the true but unknown distribution μ , i.e. $\mu \in \mathcal{M}$, and $x_{1:\infty}$ is sampled from μ . We abbreviate expectations w.r.t. μ by $\mathbf{E}[\cdot]$; for instance, $\mathbf{E}[f(x_{1:n})] = \sum_{x_{1:n} \in \mathcal{X}^n} \mu(x_{1:n}) f(x_{1:n})$. We use the (total) relative entropy D_n and squared Euclidian distance S_n to measure the distance between μ and ξ :

$$D_n := \mathbf{E} \left[\ln \frac{\mu(x_{1:n})}{\xi(x_{1:n})} \right], \quad S_n := \sum_{t=1}^n \mathbf{E} \left[\sum_{x'_t \in \mathcal{X}} \left(\mu(x'_t | x_{<t}) - \xi(x'_t | x_{<t}) \right)^2 \right]. \quad (1.5)$$

The following sequence of inequalities can be shown, which generalize Solomonoff's result (1.2): $S_n \leq D_n \leq \ln w_\mu^{-1} < \infty$. The finiteness of S_∞ implies $\xi(x'_t | x_{<t}) - \mu(x'_t | x_{<t}) \rightarrow 0$ for $t \rightarrow \infty$ w.μ.p.1 for any x'_t ($\sum_{t=1}^\infty s_t^2 < \infty \Rightarrow s_t \rightarrow 0$). We also show that $\sum_{t=1}^n \mathbf{E}[(\sqrt{\xi(x_t | x_{<t})/\mu(x_t | x_{<t})} - 1)^2] \leq D_n \leq \ln w_\mu^{-1} < \infty$, which implies $\xi(x_t | x_{<t})/\mu(x_t | x_{<t}) \rightarrow 1$ for $t \rightarrow \infty$ w.μ.p.1. This convergence motivates the belief that predictions based on (the known) ξ are asymptotically as good as predictions based on (the unknown) μ , with rapid convergence.

1.3.2 Loss Bounds

Most predictions are eventually used as a basis for some decision or action, which itself leads to some reward or loss. Let $\ell_{x_t y_t} \in [0, 1] \subset \mathbb{R}$ be the received loss when performing prediction/decision/action $y_t \in \mathcal{Y}$, and $x_t \in \mathcal{X}$ is the t^{th} symbol of the sequence. Let $y_t^A \in \mathcal{Y}$ be the prediction of a (causal) prediction scheme A . The true probability of the next symbol being x_t , given $x_{<t}$, is $\mu(x_t | x_{<t})$. The expected loss when predicting y_t is $\mathbf{E}[\ell_{x_t y_t}]$. The total μ -expected loss suffered by the A scheme in the first n predictions is

$$L_n^A := \sum_{t=1}^n \mathbf{E}[\ell_{x_t y_t^A}].$$

The goal is to minimize the expected loss. More generally, we define the A_ρ sequence prediction scheme (later also called $\text{SP}\rho$) $y_t^{A_\rho} := \arg\min_{y_t \in \mathcal{Y}} \sum_{x_t} \rho(x_t | x_{<t}) \ell_{x_t y_t}$, which minimizes the ρ -expected loss. If μ is known, A_μ is obviously the best prediction scheme in the sense of achieving minimal expected loss ($L_n^{A_\mu} \leq L_n^A$ for any A). We prove the following loss bound for the universal A_ξ predictor

$$0 \leq L_n^{A_\xi} - L_n^{A_\mu} \leq D_n + \sqrt{4L_n^{A_\mu} D_n + D_n^2} \leq 2D_n + 2\sqrt{L_n^{A_\mu} D_n}. \quad (1.6)$$

Together with $L_n \leq n$ and $D_\infty \leq \ln w_\mu^{-1} < \infty$, this shows that $\frac{1}{n} L_n^{A_\xi} - \frac{1}{n} L_n^{A_\mu} = O(n^{-1/2})$, i.e. asymptotically A_ξ achieves the optimal average loss of A_μ with rapid convergence. Moreover, $L_\infty^{A_\xi}$ is finite if $L_\infty^{A_\mu}$ is finite, and $L_n^{A_\xi}/L_n^{A_\mu} \rightarrow 1$ if $L_\infty^{A_\mu}$ is not finite. Bound (1.6) also implies $L_n^A \geq L_n^{A_\xi} - 2\sqrt{L_n^{A_\xi} D_n}$, which shows that *no* (causal) predictor A whatsoever achieves significantly less (expected) loss than A_ξ . Note that for $w_\nu = 2^{-K(\nu)}$, $D_n \leq \ln 2 \cdot K(\mu)$ is of “reasonable” size. Instantaneous loss bounds can also be proven.

1.3.3 Optimality Properties

For any predictor A , a worst-case lower bound that asymptotically matches the upper bound (1.6) can be derived. More precisely, let A be any deterministic predictor not knowing from which distribution $\mu \in \mathcal{M}$ the observed sequence $x_1 x_2 \dots$ is sampled. Predictor A knows (depends on) \mathcal{M} , w_ν , and ℓ , and has at time t access to the previous outcomes $x_{<t}$. Then for every n there is an \mathcal{M} and $\mu \in \mathcal{M}$ and ℓ and weights w_ν such that

$$L_n^A - L_n^{A_\mu} \geq \frac{1}{2} [S_n + \sqrt{4L_n^{A_\mu} S_n + S_n^2}], \quad \text{and} \quad D_n/S_n \rightarrow 1 \quad \text{for} \quad n \rightarrow \infty.$$

For the universal predictor $A = A_\xi$, the lower bound holds even without the factor $\frac{1}{2}$. This shows that bound (1.6) is quite tight in the sense that no other predictor can lead to significantly smaller bounds without making extra assumptions on \mathcal{M} , w_ν , or ℓ . For instance, for logarithmic and quadratic loss functions the regret $L_\infty^{A_\xi} - L_\infty^{A_\mu}$ is finite and bounded by $\ln w_\mu^{-1}$.

A different kind of optimality is *Pareto optimality*. Let $\mathcal{F}(\mu, \rho)$ be any performance measure of ρ relative to μ . The universal prior ξ is called Pareto optimal w.r.t. \mathcal{F} if there is no ρ with $\mathcal{F}(\nu, \rho) \leq \mathcal{F}(\nu, \xi)$ for all $\nu \in \mathcal{M}$ and strict inequality for at least one ν . We show that the universal prior ξ is Pareto optimal w.r.t. the squared distance S_n , the relative entropy D_n , and the losses L_n . That is, for all performance measures that are relevant from a decision-theoretic point of view (i.e. for all loss functions ℓ) any improvement achieved by some predictor A_ρ over A_ξ in some environments ν is balanced by a deterioration in other environments. There are non-decision-theoretic performance measures w.r.t. which ξ is *not* Pareto optimal. Pareto optimality is a rather weak notion of optimality, but it emphasizes the distinctiveness of Bayes mixture strategies.

Pareto optimality of ξ still leaves open the question of how to choose the class \mathcal{M} and the weights w_ν . We have argued that \mathcal{M}_U is the largest \mathcal{M} suitable from a computational point of view. \mathcal{M}_U is also sufficiently large if we make the mild assumption that strings are sampled from a computable probability distribution. We show that within the class of enumerable weight functions with short program, the universal weights $w_\nu = 2^{-K(\nu)}$ lead to the smallest performance bounds within an additive (to $\ln w_\nu^{-1}$) constant in all enumerable environments. This argument justifies the selection of Solomonoff-Levin's prior (1.3) among all possible Bayes mixtures.⁴

1.3.4 Miscellaneous

Games of chance. The general loss bound (1.6) can, for instance, be used to estimate the time needed to reach the winning threshold in a game of chance (defined as a sequence of bets, observations and rewards). At time t we bet, depending on the history $x_{<t}$, a certain amount of money s_t , take some action y_t , observe outcome x_t , and receive reward r_t . Our net profit, which we want to maximize, is $p_t = r_t - s_t \in [p_{\max} - p_\Delta, p_{\max}]$. The loss, which we want to minimize, can be identified with the negative (scaled) profit, $\ell_{x_k y_t} = (p_{\max} - p_t)/p_\Delta \in [0, 1]$. The A_ρ -system acts as to maximize the ρ -expected profit. Let \bar{p}_n^{ρ} be the average expected profit of the first n rounds. Bound (1.6) shows that the average profit of the A_ξ system converges to the best possible average profit \bar{p}_n^{μ} achieved by the A_μ scheme ($\bar{p}_n^{\xi} - \bar{p}_n^{\mu} = O(n^{-1/2}) \rightarrow 0$ for $n \rightarrow \infty$). If there is a profitable scheme at all, then asymptotically the universal A_ξ scheme will also become profitable with the same average profit. We further show using ξ_U that $(2p_\Delta/\bar{p}_n^{\mu})^2 \cdot \ln 2 \cdot K(\mu)$ is an upper bound on the number of bets n needed to reach the winning zone. The bound is proportional to the complexity of the environment μ .

Continuous probability classes \mathcal{M} . We have considered thus far countable probability classes \mathcal{M} , which makes sense from a computational point of view. On the other hand, in statistical parameter estimation one often has a continuous hypothesis class (e.g. a Bernoulli(θ) process with unknown $\theta \in [0, 1]$). Let $\mathcal{M} := \{\mu_\theta : \theta \in \Theta \subseteq \mathbb{R}^d\}$ be a family of probability distributions parameterized by a d -dimensional continuous parameter θ . Let $\mu \equiv \mu_{\theta_0} \in \mathcal{M}$ be the true generating distribution. For a continuous weight density $w(\theta) > 0$ the sums in (1.4) are naturally replaced by integrals: $\xi(x_{1:n}) := \int_\Theta w(\theta) \cdot \mu_\theta(x_{1:n}) d\theta$ with $\int_\Theta w(\theta) d\theta = 1$. The most important property of ξ in the discrete case was the dominance $\xi(x_{1:n}) \geq w_\nu \cdot \nu(x_{1:n})$, which was obtained from (1.4) by dropping the sum over ν . The analogous construction here is to restrict the integral over Θ to a small vicinity N_δ of θ . For sufficiently smooth μ_θ and $w(\theta)$ we expect $\xi(x_{1:n}) \gtrsim |N_{\delta_n}| \cdot w(\theta) \cdot \mu_\theta(x_{1:n})$, where $|N_{\delta_n}|$ is the volume of

⁴ Readers who smell some free lunch here [WM97] should appease their hunger with Section 3.6.5.

N_{δ_n} . This in turn leads to $D_n \lesssim \ln w_\mu^{-1} + \ln |N_{\delta_n}|^{-1}$, where $w_\mu := w(\theta_0)$. N_{δ_n} should be the largest possible region in which $\ln \mu_\theta$ is approximately flat on average. More precisely, generalizing [CB90] to the non-i.i.d. case, we show $D_n \leq \ln w_\mu^{-1} + \frac{d}{2} \ln \frac{n}{2\pi} + O(1)$, where the $O(1)$ term depends on the smoothness of μ_θ , measured by the Fisher information. D_n is no longer bounded by a constant, but still grows only logarithmically with n , the intuitive reason being the necessity to describe θ to an accuracy $O(n^{-1/2})$. So, bound (1.6) is also applicable to the case of continuously parameterized probability classes.

1.4 Rational Agents in Known Probabilistic Environments

1.4.1 The Agent Model

A very general framework for intelligent systems is that of rational agents [RN95]. In cycle k , an agent performs *action* $y_k \in \mathcal{Y}$ (output), which results in a *perception* $x_k \in \mathcal{X}$ (input), followed by cycle $k+1$, and so on. We assume that the action and perception spaces \mathcal{X} and \mathcal{Y} are finite. We write $p(x_{<k}) = y_{1:k}$ to denote the output $y_{1:k}$ of the agent's policy p on input $x_{<k}$, and similarly $q(y_{1:k}) = x_{1:k}$ for the environment q in the case of deterministic environments. We call policy p and environment q behaving in this way *chronological*. The figure on the book cover and on page 128 depicts this interaction in the case where p and q are modeled by Turing machines. Note that policy and environment are allowed to depend on the complete history. We do not make any MDP or POMDP assumption here, and we do not talk about states of the environment, only about observations. In the more general case of a *probabilistic environment*, given the history $\mathcal{Y}_{<k} y_k \equiv \mathcal{Y}_1 \dots \mathcal{Y}_{k-1} y_k \equiv y_1 x_1 \dots y_{k-1} x_{k-1} y_k$, the probability that the environment leads to perception x_k in cycle k is (by definition) $\mu(\mathcal{Y}_{<k} \underline{x}_k)$. The underlined argument \underline{x}_k in μ is a random variable, and the other non-underlined arguments $\mathcal{Y}_{<k} y_k$ represent conditions.⁵ We call probability distributions like μ *chronological*. Since value-optimizing policies (see below) can always be chosen deterministic, there is no real need to generalize the setting to probabilistic policies.

1.4.2 Value Functions & Optimal Policies

The goal of the agent is to maximize future *rewards*, which are provided by the environment through the inputs x_k . The inputs $x_k \equiv r_k o_k$ are divided into a regular part o_k and some (possibly empty or delayed) reward $r_k \in [0, r_{max}]$.⁶ We use the abbreviation

⁵ The standard notation $\mu(x_k | \mathcal{Y}_{<k} y_k)$ for conditional probabilities destroys the chronological order and would become confusing in later expressions.

⁶ In the reinforcement learning literature when dealing with (PO)MDPs the reward is usually considered to be a function of the environmental state. The zero-

$$\mu(y_{<k}y_{k:m}) = \mu(y_{<k}y_k) \cdot \mu(y_{1:k}y_{k+1}) \cdot \dots \cdot \mu(y_{<m}y_m),$$

which is essentially the chain rule, and $\epsilon = y_{<1}$ for the empty string. We define the (total) *value* of policy p in environment μ , or shorter, the μ -value of p , as the μ -expected reward sum

$$V_\mu^p := \sum_{x_{1:m}} (r_1 + \dots + r_m) \mu(y_{1:m})_{|y_{1:m}=p(x_{<m})}, \quad (1.7)$$

where m is the *lifespan* or initial *horizon* of the agent. The optimal policy p^μ that maximizes the value V_μ^p is

$$p^\mu := \arg \max_p V_\mu^p, \quad V_\mu^* := V_\mu^{p^\mu} = \max_p V_\mu^p \geq V_\mu^p \quad \forall p.$$

The policy p^μ , which we call *AI μ model*, is optimal in the sense that no other policy for an agent leads to higher μ -expected reward. Explicit expressions for the action y_k in cycle k of the μ -optimal policy p^μ and their value V_μ^* are

$$y_k = y_k^\mu := \arg \max_{y_k} \sum_{x_k} \max_{y_{k+1}} \sum_{x_{k+1}} \dots \max_{y_m} \sum_{x_m} (r_k + \dots + r_m) \cdot \mu(y_{<k}y_{k:m}), \quad (1.8)$$

$$V_\mu^* = \max_{y_1} \sum_{x_1} \max_{y_2} \sum_{x_2} \dots \max_{y_m} \sum_{x_m} (r_1 + \dots + r_m) \cdot \mu(y_{1:m}), \quad (1.9)$$

where $y_{<k}$ is the actual history. We show that these definitions are consistent and correctly capture our intention. For instance, consider the *expectimax* expression (1.9): The best expected reward is obtained by averaging over possible perceptions x_i and by maximizing over the possible actions y_i . This has to be done in chronological order $y_1 x_1 \dots y_m x_m$ to correctly incorporate the dependencies of x_i and y_i on the history. This is the origin of the alternating *expectimax* sequence, which is similar to the well-known minimax sequence/tree/algorithm in game theory.

1.4.3 Sequential Decision Theory & Reinforcement Learning

One can relate (1.9) to the Bellman equations [Bel57] of sequential decision theory by identifying complete histories $y_{<k}$ with states, $\mu(y_{<k}y_k)$ with the state transition matrix, V_μ^* with the value function, and y_k with the action in cycle k [BT96, RN95]. Due to the use of complete histories as state space, the AI μ model assumes neither stationarity nor the Markov property nor complete accessibility of the environment. Every state occurs at most once in the lifetime of the system. For this and other reasons the explicit formulation (1.8) is more

assumption analogue here is that the reward r_k is some probabilistic function μ' depending on the complete history. It is very convenient to integrate r_k into x_k and μ' into μ .

natural and useful here than to enforce a pseudo-recursive Bellman equation form.

As we have in mind a universal system with complex interactions, the action and perception spaces \mathcal{Y} and \mathcal{X} are huge (e.g. video images), and every action or perception itself occurs usually only once in the lifespan m of the agent. As there is no (obvious) universal similarity relation on the state space, an effective reduction of its size is impossible, but there is no principle problem in determining y_k from (1.8) as long as μ is known and computable and \mathcal{X} , \mathcal{Y} and m are finite.

Things drastically change if μ is unknown. Reinforcement learning algorithms [BT96, KLM96, SB98] are commonly used in this case to learn the unknown μ or directly its value. They succeed if the state space is either small or has effectively been made small by generalization or function approximation techniques. In any case, the solutions are either ad hoc, work in restricted domains only, have serious problems with state space exploration versus exploitation, are prone to diverge, or have nonoptimal learning rates. There is no universal and optimal solution to this problem so far. The central theme of this book is to present a new model and to argue that it formally solves all these problems in an optimal way. The true probability distribution μ will not be learned directly, but will be replaced by some generalized universal prior ξ_U , which converges to μ , similarly to the induction (SP) case.

1.5 The Universal Algorithmic Agent AIXI

1.5.1 The Universal AIXI Model

We have developed enough formalism to present the universal AIXI model. All we have to do is to suitably generalize Solomonoff's universal prior M and to replace the true but unknown probability μ in the $AI\mu$ model by this generalized M . Similarly to (1.1), we define M as the $2^{-\ell(q)}$ weighted sum over all chronological programs (environments) q that output $x_{1:k}$, but with $y_{1:k}$ provided on the input tape. This also generalizes ξ_U (within an irrelevant multiplicative constant):

$$\xi(\underline{y}_{1:k}) = \xi_U(\underline{y}_{1:k}) \stackrel{\propto}{=} M(\underline{y}_{1:k}) := \sum_{q: q(y_{1:k})=x_{1:k}} 2^{-\ell(q)}. \quad (1.10)$$

If not clear from context, we add superscripts SP and AI to ξ , to resolve ambiguities between (1.3) and (1.10). Replacing μ by ξ in (1.8) the *AIXI system* outputs

$$y_k = y_k^\xi := \arg \max_{y_k} \sum_{x_k} \dots \max_{y_m} \sum_{x_m} (r_k + \dots + r_m) \cdot \xi(y_{<k} \underline{y}_{k:m}) \quad (1.11)$$

in cycle k given the history $y_{<k}$. The ξ -value V_ξ^p and the universal value V_ξ^* are defined as in (1.7) and (1.9), with μ replaced by ξ . The AIXI model and

its behavior is completely defined by (1.10) and (1.11). It (slightly) depends on the choice of the universal Turing machine, because $K()$ and $\ell()$ depend on U and hence are defined only up to terms of order one. The AIXI model also depends on the choice of \mathcal{X} and \mathcal{Y} , but we do not expect any bias when the spaces are chosen sufficiently large and simple, e.g. all strings of length 2^{16} . Choosing \mathcal{N} as the I/O spaces would be ideal, but whether the maxima (or suprema) exist in this case has to be shown beforehand. The only nontrivial dependence is on the horizon m . Ideally, we would like to choose $m = \infty$, but there are several subtleties to be unraveled later, which prevent at least a naive limit $m \rightarrow \infty$. So apart from m and unimportant details, *the AIXI system is uniquely defined by (1.10) and (1.11) without adjustable parameters.*

1.5.2 On the Optimality of AIXI

Universality and convergence of ξ . One can show that also ξ defined in (1.10) is universal and rapidly converges to μ analogous to the induction (SP) case. If we take a finite product of conditional ξ 's and use the chain rule, we see that also $\xi(\mathbf{x}_{<k} \mathbf{y}_{k:k+h})$ converges to $\mu(\mathbf{x}_{<k} \mathbf{y}_{k:k+h})$ for $k \rightarrow \infty$. This gives confidence that the outputs y_k^ξ of the AIXI model (1.11) could converge to the outputs y_k^μ of the $\text{AI}\mu$ model (1.8), at least for a bounded moving horizon h . The problems with a fixed horizon m and especially $m \rightarrow \infty$ will be discussed at the end of this section.

Universally optimal AI systems. We call an AI model *universal* if it is independent of the true environment μ (unbiased, model-free) and is able to solve any solvable problem and learn any learnable task. Further, we call a universal model *universally optimal* if there is no program that can solve or learn significantly faster (in terms of interaction cycles). As the AIXI model is parameter-free, ξ converges to μ , the $\text{AI}\mu$ model is itself optimal, and we expect no other model to converge faster to $\text{AI}\mu$ by analogy to the SP case,

we expect AIXI to be universally optimal.

This is our main claim. Further support is given below.

Intelligence order relation. We want to call a policy p *more or equally intelligent* than a policy p' and write $p \succeq p'$ if p yields in every cycle k and for every fixed history $\mathbf{x}_{<k}$ higher (future) ξ -expected reward sum than p' . It is a formal exercise to show that $p^\xi \succeq p$ for all p . The AIXI model is hence the most intelligent agent w.r.t. \succeq . Relation \succeq is a universal order relation in the sense that it is free of any parameters (except m) or specific assumptions about the environment. A proof that \succeq is a reasonable intelligence order (which we believe to be true) would prove that AIXI is universally optimal.

Value bounds. The values V_ρ^* associated with the $\text{AI}\rho$ systems correspond roughly to the negative total loss $-L_n^{\Lambda_\rho}$ (with $n=m$) of the $\text{SP}\rho (= \Lambda_\rho)$ systems.

In the SP case we were interested in small bounds for the regret $L_n^{A_\xi} - L_n^{A_\mu}$. Unfortunately, simple value bounds for AIXI or any other AI system in terms of V_ν^* analogous to the loss bound (1.6) cannot hold. We even have difficulties in specifying what we can expect to hold for AIXI or any AI system that claims to be universally optimal. In SP, the only important property of μ for proving loss bounds was its complexity $K(\mu)$. In the AI case, there are no useful bounds in terms of $K(\mu)$ only. We either have to study restricted problem or environmental classes or consider bounds depending on other properties of μ , rather than on its complexity only.

1.5.3 Value-Related Optimality Results

The mixture distribution ξ . In the following, we consider general Bayes mixtures ξ over classes \mathcal{M} of chronological probability distributions ν :

$$\xi(\underline{y}_{1:m}) = \sum_{\nu \in \mathcal{M}} w_\nu \nu(\underline{y}_{1:m}) \quad \text{with} \quad \sum_{\nu \in \mathcal{M}} w_\nu = 1 \quad \text{and} \quad w_\nu > 0 \quad \forall \nu \in \mathcal{M}.$$

We define V_ξ^p , p^ξ , and V_ξ^* as in (1.7)–(1.9) with μ replaced by ξ . Policy p^ξ is called the $\text{AI}\xi$ model. For $\xi = \xi_U$ the $\text{AIXI} \equiv \text{AI}\xi_U$ model is recovered. If μ is unknown, but known to belong to the known class \mathcal{M} , it is natural to follow policy p^ξ , which maximizes V_ξ^p . The (true μ -)expected reward when following policy p^ξ is $V_\mu^{p^\xi}$. The optimal (but infeasible) policy p^μ yields reward $V_\mu^{p^\mu} \equiv V_\mu^*$. It is now of interest (a) whether there are policies with uniformly larger value than $V_\mu^{p^\xi}$ and (b) how close $V_\mu^{p^\xi}$ is to V_μ^* .

Linearity and convexity of V_ρ in ρ . The following properties of V_ρ are crucial. V_ρ^p is a linear function in ρ , and V_ρ^* is a convex function in ρ in the sense that

$$V_\xi^p = \sum_{\nu \in \mathcal{M}} w_\nu V_\nu^p \quad \text{and} \quad V_\xi^* \leq \sum_{\nu \in \mathcal{M}} w_\nu V_\nu^*.$$

Linearity is obvious from the definition of V_ρ^p , and convexity follows easily from the convexity of \max_p and nonnegativity of the weights w_ν . One loose interpretation of the convexity is that a mixture can never increase performance.

Pareto optimality of $\text{AI}\xi$. Similarly to the SP case, one can show that p^ξ is *Pareto optimal* in the sense that there is no other policy p with $V_\nu^p \geq V_\nu^{p^\xi}$ for all $\nu \in \mathcal{M}$ and strict inequality for at least one ν . In particular, AIXI is Pareto optimal.

Self-optimizing policy p^ξ w.r.t. average value. Since we do not know the true environment μ in advance, we are interested under which circumstances⁷

⁷ Here and elsewhere we interpret $a_m \rightarrow b_m$ as an abbreviation for $a_m - b_m \rightarrow 0$. $\lim_{m \rightarrow \infty} b_m$ may not exist.

$$\frac{1}{m} V_{\nu}^{p^{\xi}} \rightarrow \frac{1}{m} V_{\nu}^{*} \quad \text{for horizon } m \rightarrow \infty \quad \text{for all } \nu \in \mathcal{M}. \quad (1.12)$$

Note that V_{ν} as well as $p^{\xi} = p_m^{\xi}$ depend on m . The least we must demand from \mathcal{M} to have a chance that (1.12) is true is that there exists a policy (sequence) $\tilde{p} = \tilde{p}_m$ at all with this property, i.e.

$$\exists \tilde{p} : \frac{1}{m} V_{\nu}^{\tilde{p}} \rightarrow \frac{1}{m} V_{\nu}^{*} \quad \text{for horizon } m \rightarrow \infty \quad \text{for all } \nu \in \mathcal{M}. \quad (1.13)$$

We show that this necessary condition is also sufficient, i.e. (1.13) implies (1.12). This is another (asymptotic) optimality property of policy p^{ξ} . If universal convergence in the sense of (1.13) is possible at all in a class of environments \mathcal{M} , then policy p^{ξ} converges in the same sense (1.12). We call policies \tilde{p} with a property like (1.13) *self-optimizing* [KV86].

Unfortunately, the result is not an asymptotic convergence statement of a single policy p^{ξ} , since p^{ξ} depends on m . The result merely says that under the stated conditions the average value of p_m^{ξ} is arbitrarily close to optimum for sufficiently large (pre-chosen) horizon m . This weakness will be resolved in the following.

Discounted future value function. We now shift our focus from the total value to future values (value-to-go). First, we have to get rid of the horizon parameter m . We eliminate the horizon by discounting the rewards $r_k \leadsto \gamma_k r_k$ with $\gamma_k \geq 0$ and $\sum_{i=1}^{\infty} \gamma_i < \infty$ and taking $m \rightarrow \infty$. The analogue of m is now an effective horizon h_k^{eff} , which may be defined by $\sum_{i=k}^{k+h_k^{\text{eff}}} \gamma_i \approx \sum_{i=k+h_k^{\text{eff}}}^{\infty} \gamma_i$. Furthermore, we renormalize the value V by $\sum_{i=k}^{\infty} \gamma_i$ and denote it by $V_{k\gamma}$. Finally, we extend the definition to probabilistic policies π (which is not essential). We define the γ -discounted weighted-average future *value* of (probabilistic) policy π in environment ρ given history $\underline{y}_{<k}$, or shorter, the ρ -value of π given $\underline{y}_{<k}$, as

$$V_{k\gamma}^{\pi\rho}(\underline{y}_{<k}) := \frac{1}{\Gamma_k} \lim_{m \rightarrow \infty} \sum_{\underline{y}_{k:m}} (\gamma_k r_k + \dots + \gamma_m r_m) \rho(\underline{y}_{<k} \underline{y}_{k:m}) \pi(\underline{y}_{<k} \underline{y}_{k:m}),$$

with $\Gamma_k := \sum_{i=k}^{\infty} \gamma_i$. The policy p^{ρ} is defined as to maximize the future value $V_{k\gamma}^{\pi\rho}$:

$$p^{\rho} := \arg \max_{\pi} V_{k\gamma}^{\pi\rho}, \quad V_{k\gamma}^{*\rho} := V_{k\gamma}^{p^{\rho}\rho} = \max_{\pi} V_{k\gamma}^{\pi\rho} \geq V_{k\gamma}^{\pi\rho} \forall \pi.$$

Setting $\gamma_k = 1$ for $k \leq m$ and $\gamma_k = 0$ for $k > m$ gives back the old undiscounted model with horizon m and $V_{1\gamma}^{\rho} = \frac{1}{m} V_{\rho}^p$. Note that $V_{k\gamma}$ depends on the realized history $\underline{y}_{<k}$. More important, p^{ρ} can be shown to be independent of k . Similarly to the undiscounted case, one can prove that for every k and history $\underline{y}_{<k}$, $V_{k\gamma}^{\pi\rho}$ is a linear function in ρ , $V_{k\gamma}^{*\rho}$ is a convex function in ρ , and p^{ξ} is Pareto optimal in the sense that there is no other policy π with $V_{k\gamma}^{\pi\nu} \geq V_{k\gamma}^{p^{\xi}\nu}$ for all $\nu \in \mathcal{M}$ and strict inequality for at least one ν . Finally, p^{ξ} is self-optimizing (w.r.t. discounted value) if \mathcal{M} admits self-optimizing policies:

$$\text{If } \exists \tilde{\pi} \forall \nu : V_{k\gamma}^{\tilde{\pi}\nu} \xrightarrow{k \rightarrow \infty} V_{k\gamma}^{*\nu} \text{ w.}\nu.\text{p.1} \implies V_{k\gamma}^{p^\xi \mu} \xrightarrow{k \rightarrow \infty} V_{k\gamma}^{*\mu} \text{ w.}\mu.\text{p.1}.$$

The probability qualifier refers to the historic perceptions $x_{<k}$. The historic actions $y_{<k}$ are arbitrary. Note that k is a real running value, namely the current cycle number, whereas m was a pre-chosen fixed horizon.

1.5.4 Markov Decision Processes

From all possible environments, Markov (decision) processes are probably the most intensively studied ones. μ is called a (completely observable stationary) *Markov decision process* (MDP) if the probability of perceiving $x_k \in \mathcal{X}$, given history $\mathbf{x}_{<k} y_k$ only depends on the last action $y_k \in \mathcal{Y}$ and the last perception x_{k-1} , i.e. if $\mu(\mathbf{x}_{<k} y_k \underline{x}_k) = \mu(x_{k-1} y_k \underline{x}_k)$. In this case x_k is called a *state*, \mathcal{X} the *state space*, and $\mu(x_{k-1} y_k \underline{x}_k)$ the *transition matrix*. An MDP μ is called *ergodic* if there exists a policy under which every state is visited infinitely often with probability 1. If an MDP $\mu(x_{k-1} y_k \underline{x}_k)$ is independent of the action y_k it is a *Markov process*; if it is independent of the last perception x_{k-1} it is an *i.i.d.* process.

Stationary MDPs μ with geometric discounting $\gamma_k = \gamma^k$ have stationary optimal policies p^μ mapping the same state/perception x_k always to the same action y_k . On the other hand, a mixture ξ of MDPs is itself not an MDP, i.e. $\xi \notin \mathcal{M}_{\text{MDP}}$, which implies that p^ξ is, in general, not a stationary policy.

One can construct self-optimizing policies for the class of ergodic MDPs w.r.t. the average value $\frac{1}{m} V_\rho^{\pi\rho}$ and if $\frac{\gamma_{k+1}}{\gamma_k} \rightarrow 1$ also w.r.t. to the discounted future value $V_{k\gamma}^{\pi\rho}$. The necessary condition $\frac{\gamma_{k+1}}{\gamma_k} \rightarrow 1$ ensures unboundedly increasing effective horizon h_k^{eff} . The existence of self-optimizing policies for ergodic MDPs implies that for a countable class \mathcal{M} of ergodic MDPs, the policies p_m^ξ maximizing V_ξ^p and p^ξ maximizing $V_{k\gamma}^{\pi\xi}$ are self-optimizing in the sense that

$$\forall \nu \in \mathcal{M} : \frac{1}{m} V_{1m}^{p_m^\xi \nu} \xrightarrow{m \rightarrow \infty} \frac{1}{m} V_{1m}^{*\nu} \quad \text{and} \quad V_{k\gamma}^{p^\xi \nu} \xrightarrow{k \rightarrow \infty} V_{k\gamma}^{*\nu} \quad \text{if} \quad \frac{\gamma_{k+1}}{\gamma_k} \rightarrow 1. \quad (1.14)$$

We also show that if \mathcal{M} is finite, then the speed of the first convergence is at least $O(m^{-1/3})$. The conditions $\Gamma_k < \infty$ and $\frac{\gamma_{k+1}}{\gamma_k} \rightarrow 1$ on the discount sequence are, for instance, satisfied for $\gamma_k = 1/k^2$, but *not* for the popular geometric discount $\gamma_k = \gamma^k$, which has finite effective horizon.

Limits (1.14) show that p^ξ is self-optimizing for bandits, i.i.d. processes, and classification tasks, since they are special (degenerate) cases of ergodic MDPs. The existence of self-optimizing policies is not limited to (subclasses of ergodic) MDPs. Certain classes of POMDPs, k^{th} -order ergodic MDPs, factorizable environments, repeated games, and prediction problems are not MDPs, but nevertheless admit self-optimizing policies. Hence the corresponding Bayes optimal mixture policy p^ξ is self-optimizing.

1.5.5 The Choice of the Horizon

The only significant arbitrariness in the AIXI model lies in the choice of the lifespan m or in the discounted case in the discount sequence γ_k . We will not discuss ad hoc choices for specific problems. We are interested in universal choices. In many cases the time we are willing to run a system depends on the quality of its actions. Hence, the lifetime, if finite at all, is not known in advance. Geometric discounting $r_k \rightsquigarrow r_k \cdot \gamma^k$ solves the mathematical problem of $m \rightarrow \infty$ but is not a real solution, since an effective horizon $h^{eff} \sim \ln \gamma^{-1} < \infty$ has been introduced. The scale-invariant discounting $r_k \rightsquigarrow r_k \cdot k^{-\alpha}$ with $\alpha > 1$ has a dynamic horizon $h \sim k$. This choice has some appeal, as it seems that humans of age k years also usually do not plan their lives for more than the next $\sim k$ years. It also satisfies the condition $\frac{\gamma_{k+1}}{\gamma_k} \rightarrow 1$, necessary for AI ξ being self-optimizing in ergodic MDPs. The largest lower semicomputable horizon with guaranteed finite reward sum $\Gamma_1 < \infty$ is obtained by the discount $r_k \rightsquigarrow r_k \cdot 2^{-K(k)}$, where $K(k)$ is the Kolmogorov complexity of k . This is maybe the most attractive universal discount. It is similar to a near-harmonic discount $r_k \rightsquigarrow r_k \cdot k^{-(1+\varepsilon)}$, since $2^{-K(k)} \leq 1/k$ for most k and $2^{-K(k)} \geq c/(k \log^2 k)$ for some constant c . We are not sure whether the choice of the horizon is of marginal importance, as long as it is chosen sufficiently large, or whether the choice will turn out to be a central topic for the AIXI model or for the planning aspect of any universal AI system in general. Most, if not all, problems in agent design of balancing exploration and exploitation vanish by a sufficiently large choice of the (effective) horizon and a sufficiently general prior.

1.6 Important Environmental Classes

In this and the next section we define $\xi = \xi_U \stackrel{\times}{=} M$ be Solomonoff's prior, i.e. AI ξ =AIXI. Each subsection represents an abstract on what will be done in the corresponding section of Chapter 6.

1.6.1 Introduction

In order to give further support for the universality and optimality of the AI ξ theory, we apply AI ξ to a number of problem classes. They include sequence prediction, strategic games, function minimization and, especially, how AI ξ learns to learn supervised. For some classes we give concrete examples to illuminate the scope of the problem class. We first formulate each problem class in its natural way (when μ^{problem} is known) and then construct a formulation within the AI μ model and prove its equivalence. We then consider the consequences of replacing μ by ξ . The main goal is to understand why and how the problems are solved by AI ξ . We only highlight special aspects of each problem class. The goal is to give a better picture of the flexibility of the AI ξ model.

1.6.2 Sequence Prediction (SP)

Using the $\text{AI}\mu$ model for sequence prediction (SP) is identical to Bayesian sequence prediction $\text{SP}\mu$. One might expect, when using the $\text{AI}\xi$ model for sequence prediction, one would recover exactly the universal sequence prediction scheme $\text{SP}\xi$, as $\text{AI}\xi$ was a unification of the $\text{AI}\mu$ model and the idea of universal probability ξ . Unfortunately, this is not the case. One reason is that ξ is only a probability distribution in the inputs x and not in the outputs y . This is also one of the origins of the difficulty of proving loss/value bounds for $\text{AI}\xi$. Nevertheless, we argue that $\text{AI}\xi$ is as well suited for sequence prediction as $\text{SP}\xi$. In a very limited setting we prove a (weak) error bound for $\text{AI}\xi$, which gives hope that a general proof is attainable.

1.6.3 Strategic Games (SG)

A very important class of problems are strategic games (SG). We restrict ourselves to deterministic strictly competitive strategic games like chess. If the environment is a minimax player, the $\text{AI}\mu$ model itself reduces to a minimax strategy. Repeated games of fixed lengths are a special case of factorizable μ . The consequences of variable game lengths are sketched. The $\text{AI}\xi$ model has to learn the rules of the game under consideration, as it has no prior information about these rules. We describe how $\text{AI}\xi$ actually learns these rules.

1.6.4 Function Minimization (FM)

Many problems fall into the category ‘resource-bounded function minimization’ (FM). They include the traveling salesman problem, minimizing production costs, inventing new materials or even producing, e.g. nice paintings, which are (subjectively) judged by a human. The task is to (approximately) minimize some function $f:\mathcal{Y}\rightarrow\mathcal{Z}$ within a minimal number of function calls. We will see that a greedy model trying to minimize f in every cycle fails. Although the greedy model has nothing to do with downhill or gradient techniques (there is nothing like a gradient or direction for functions over \mathcal{Y}), which are known to fail, we discover the same difficulties. FM has already nearly the full complexity of general AI. The reason being that FM can actively influence the information gathering process by its trials y_k (whereas SP and CF=classification cannot). We discuss in detail the optimal $\text{FM}\mu$ model and its inventiveness in choosing the $y\in\mathcal{Y}$. A discussion of the subtleties when using $\text{AI}\xi$ for function minimization follows.

1.6.5 Supervised Learning from Examples (EX)

Reinforcement learning, as the $\text{AI}\xi$ model does, is an important learning technique, but not the only one. To improve the speed of learning, supervised learning, i.e. learning by acquiring knowledge, or learning from a constructive

teacher, is necessary. We show how AI ξ learns to learn supervised. It actually establishes supervised learning very quickly within $O(1)$ cycles.

1.6.6 Other Aspects of Intelligence

Finally, we give a brief survey of other general aspects, ideas and methods in AI, and their connection to the AI ξ model. Some aspects are directly included in the AI ξ model, while others are or should be emergent.

1.7 Computational Aspects

Up to now we have shown the universal character of the AIXI model but have completely ignored computational aspects. We start by developing an algorithm $M_{p^*}^\varepsilon$ that is capable of solving any well-defined problem p as quickly as the fastest algorithm computing a solution to p , save for a factor of $1+\varepsilon$ and lower-order additive terms. Based on a similar idea we then construct a computable version of the AIXI model.

1.7.1 The Fastest & Shortest Algorithm for All Well-Defined Problems

Introduction. A wide class of problems can be phrased in the following way. Given a formal specification $f: \mathcal{X} \rightarrow \mathcal{Y}$ of a problem depending on some parameter $x \in \mathcal{X}$, we are interested in a fast algorithm computing solution $y \in \mathcal{Y}$.

Levin search is (within a large constant factor) the fastest algorithm to invert a function $g: \mathcal{Y} \rightarrow \mathcal{X}$, if g can be evaluated quickly [Lev73b, Lev84]. Levin search can also handle time-limited optimization problems [Sol86]. Prime factorization, graph coloring, and truth assignments are example problems suitable for Levin search, if we want to find a solution, since verification is quick. Levin search cannot decide the corresponding decision problems. It is also not applicable to, e.g. matrix multiplication and reinforcement learning, since the verification task g is as hard as the computation task. Blum's speed-up theorem [Blu67, Blu71] shows that there are types of problems f for which an (incomputable) sequence of speed-improving algorithms (of increasing size) exists, but no fastest algorithm.

In the approach presented here, we consider only those algorithms that *provably* solve a given problem and have a fast (i.e. quickly computable) time bound. Neither the programs themselves nor the proofs need to be known in advance. Under these constraints we construct the asymptotically fastest algorithm save a factor of $1+\varepsilon$ that solves any well-defined problem f .

The fast algorithm $M_{p^*}^\varepsilon$. Let p^* be a given algorithm computing $p^*(x)$ from x , or, more generally, a specification of a function f . One ingredient to

our fastest algorithm $M_{p^*}^\varepsilon$ to compute $p^*(x)$ is an enumeration of proofs of increasing length in some formal axiomatic system. If a proof actually proves that some p is functionally equivalent to p^* , and p has time bound t_p , the tuple (p, t_p) is added to a list L . The program p in L with the currently smallest time bound $t_p(x)$ is executed. By construction, the result $p(x)$ is identical to $p^*(x)$. The trick to achieve a small runtime is to schedule everything in a proper way, in order not to lose too much performance by computing slow p 's and t_p 's before *the* p has been found.

More formally, we say that a program “ p computes function f ”, when a universal reference Turing machine U on input (p, x) computes $f(x)$ for all x . This is denoted by $U(p, x) = f(x)$. To be able to talk about proofs, we need a formal logic system $(\forall, \lambda, y_i, c_i, f_i, R_i, \rightarrow, \wedge, =, \dots)$ and axioms and inference rules. A proof is a sequence of formulas, where each formula is either an axiom or inferred from previous formulas in the sequence by applying the inference rules. We only need to know that *provability*, *Turing Machines*, and *computation time* can be formalized, and that the set of (correct) proofs is enumerable. We say that p is provably equivalent to p^* if the formula $[\forall y : U(p, y) = U(p^*, y)]$ can be proven. Let us fix $\varepsilon \in (0, \frac{1}{2})$. $M_{p^*}^\varepsilon$ runs three algorithms A , B , and C in parallel:

$M_{p^*}^\varepsilon(x)$

Initialize the shared variables
 $L := \{\}$, $t_{fast} := \infty$, $p_{fast} := p^*$.
 Start algorithms A , B , and C
 in parallel with relative computational
 resources ε , ε , and $1 - 2\varepsilon$, respectively.

B

Compute all $t(x)$ in parallel
 for all $(p, t) \in L$ with
 relative computation time $2^{-\ell(p) - \ell(t)}$.
if for some t , $t(x) < t_{fast}$,
then $t_{fast} := t(x)$ and $p_{fast} := p$.
continue

A

Run through all proofs.
if a proof proves for some (p, t) that
 $p(\cdot)$ is equivalent to (computes) $p^*(\cdot)$
 and has time bound $t(\cdot)$
then add (p, t) to L .

C

run U on (p_{fast}, x) .
 For each time step decrease t_{fast} by 1.
if U halts **then** print result $U(p_{fast}, x)$
 and abort computation of A , B and C .

Note that A and B only terminate when aborted by C . It is obvious that $M_{p^*}^\varepsilon$ is equivalent to (computes) p^* . We show that the computation time of $M_{p^*}^\varepsilon$ is bounded by

$$time_{M_{p^*}^\varepsilon}(x) \leq (1 + \varepsilon) \cdot t_p(x) + \frac{d_p}{\varepsilon} \cdot time_{t_p}(x) + \frac{c_p}{\varepsilon},$$

$$d_p = 3 \cdot 2^{\ell(p) + \ell(t_p)}, \quad c_p = 3 \cdot 2^{\ell(\text{proof } p) + 1} \cdot O(\ell(\text{proof } p)^2),$$

where p is any algorithm, provably computing the same function as p^* with computation time provably bounded by the function $t_p(x)$ for all x , and $time_{t_p}(x)$ is the time needed to compute the time bound $t_p(x)$. Known

time bounds for practical problems can often be computed quickly, i.e. $time_{t_p}(x)/time_p(x)$ often converges very quickly to zero. Furthermore, from a practical point of view, the provability restrictions are often rather weak. Hence, we have constructed for all those problems a solution that is asymptotically only a factor $1+\varepsilon$ slower than the (provably) fastest algorithm. On the flip side, for realistically sized problems, the lower-order terms usually dominate, which limits the practical use of $M_{p^*}^\varepsilon$.

Algorithmic complexity and the shortest algorithm. A natural definition for the (Kolmogorov) complexity of a function f is the length of the shortest program computing f : $K'(f) := \min_p \{\ell(p) : U(p, x) = f(x) \forall x\}$. Unfortunately, K' suffers from not even being approximable, since functional equality of programs is in general undecidable. Let p^* be a formal specification or a program for f . Using $K(p^*)$ is also not a suitable alternative, since it essentially depends on the choice of p^* because, e.g. “dead code” in p^* contributes to $K(p^*)$. A satisfactory solution is to take the length of the shortest program *provably* equivalent to p^* :

$$K''(p^*) := \min_p \{\ell(p) : \text{a proof of } [\forall y: U(p, y) = U(p^*, y)] \text{ exists}\}.$$

K'' (like K) is upper semicomputable. Let p' be some short description of p^* . We are now concerned with the computation time of p' . Could we get slower and slower algorithms by compressing p^* more and more? Interestingly, this is not the case. Inventing complex (long) programs is *not* necessary to construct asymptotically fast algorithms, under the stated provability assumptions, in contrast to Blum’s theorem [Blu67, Blu71]. We show that there exists a program \tilde{p} , equivalent to p^* with

$$\begin{aligned} (i) \quad \ell(\tilde{p}) &\leq K''(p^*) + O(1), \\ (ii) \quad time_{\tilde{p}}(x) &\leq (1 + \varepsilon) \cdot t_p(x) + \frac{d_p}{\varepsilon} \cdot time_{t_p}(x) + \frac{c_p}{\varepsilon}, \end{aligned}$$

where p is any program provably equivalent to p^* with computation time provably less than $t_p(x)$. That is, \tilde{p} is simultaneously among the shortest *and* fastest programs.

Generalizations. Algorithm $M_{p^*}^\varepsilon$ can be modified to handle I/O streams, definable by a Turing machine with unidirectional input and output tapes (and bidirectional work tapes) receiving an input stream and producing an output stream, as is the case in the agent setup.

1.7.2 Time-Bounded AIXI Model

The major drawback of the AIXI model is that it is uncomputable. To overcome this problem, we construct a modified algorithm $AIXI_{t,l}$, which is still superior to any other time t and length l bounded agent. The computation

time of AIXI^l is of the order $t \cdot 2^l$. Reducing the large factor 2^l along the lines of the previous subsection is possible, but will not be presented here.

Non-effectiveness of AIXI. $\xi^{\text{AI}} = \xi_U^{\text{AI}}$ is not a computable but only an enumerable semimeasure. Hence, the output \hat{y}_k of the AIXI model is only asymptotically computable (approximable). AIXI yields an algorithm that produces a sequence of trial outputs eventually converging to the correct output \hat{y}_k , but one can never be sure whether one has already reached it. Besides this, convergence is extremely slow, so this type of asymptotic computability is of no direct practical use. Furthermore, the replacement of ξ^{AI} by time-limited versions [LV91, LV97], which is suitable for sequence prediction, fails for the AIXI model. This leads to the issues addressed next.

Time bounds and effectiveness. Let \tilde{p} be a policy that calculates an acceptable output within a reasonable time \tilde{t} per interaction cycle. This sort of computability assumption, namely, that a general-purpose computer of sufficient power and appropriate program is able to behave in an intelligent way, is the very basis of AI research. Here it is not necessary to discuss what exactly is meant by ‘reasonable time/intelligence’ and ‘sufficient power’. What we are interested in is whether there is a computable version of the AIXI system that is superior or equal to any policy p with computation time per cycle of at most \tilde{t} .

What one can realistically hope to construct is an $\text{AIXI}\tilde{t}\tilde{l}$ system of computation time $c \cdot \tilde{t}$ per cycle for some constant c . The idea is to run all programs p of length $\leq \tilde{l} := \ell(\tilde{p})$ and time $\leq \tilde{t}$ per cycle and pick the best output in the sense of maximizing the *universal value* V_ξ^* . The total computation time is $c \cdot \tilde{t}$ with $c \approx 2^{\tilde{l}}$. Unfortunately, V_ξ^* cannot be used directly since this measure is itself only semicomputable and the approximation quality by using computable versions of ξ^{AI} given a time of order $c \cdot \tilde{t}$ is crude [LV97]. On the other hand, we *have* to use a measure that converges to V_ξ^* for $\tilde{t}, \tilde{l} \rightarrow \infty$, since we want the $\text{AIXI}\tilde{t}\tilde{l}$ model to converge to the AIXI model in that case.

Valid approximations. We suggest the following solution satisfying the above conditions: The main idea is to consider *extended chronological incremental policies* p , which in addition to the regular output y_k^p rate their own output with w_k^p . The $\text{AIXI}\tilde{t}\tilde{l}$ model selects the output $\hat{y}_k = y_k^p$ of the policy p with highest rating w_k^p . Policy p might suggest any output y_k^p , but it is not allowed to rate itself with an arbitrarily high w_k^p if one wants w_k^p to be a reliable criterion for selecting the best p . One must demand that no policy p is allowed to claim that it is better than it actually is. We define a logical predicate $\text{VA}(p)$, called *valid approximation*, which is true if and only if p *always* satisfies $w_k^p \leq V_\xi^p(y_{<k})$, i.e. never overrates itself. $V_\xi^p(y_{<k})$ is the ξ^{AI} -expected future reward under policy p . Valid policies p can then be (partially) ordered w.r.t. their rating w_k^p .

The universal time-bounded AIXI^l system. In the following, we describe the algorithm p^* underlying the $\text{AIXI}\tilde{t}\tilde{l}$ system. It is essentially based

on the selection of the best algorithms p_k^* out of the time \tilde{t} and length \tilde{l} bounded policies p , for which there exists a proof P of $\text{VA}(p)$ with length $\leq l_P$.

1. Create all binary strings of length l_P and interpret each as a coding of a mathematical proof in the same formal logic system in which $\text{VA}(\cdot)$ has been formulated. Take those strings that are proofs of $\text{VA}(p)$ for some p and keep the corresponding programs p .
2. Eliminate all p of length $> \tilde{l}$.
3. Modify the behavior of all remaining p in each cycle k as follows: Nothing is changed if p outputs some $w_k^p y_k^p$ within \tilde{t} time steps. Otherwise stop p and write $w_k = 0$ and some arbitrary y_k to the output tape of p . Let P be the set of all those modified programs.
4. Start first cycle: $k := 1$.
5. Run every $p \in P$ on extended input $\dot{y}\dot{x}_{<k}$, where all outputs are redirected to some auxiliary tape: $p(\dot{y}\dot{x}_{<k}) = w_1^p y_1^p \dots w_k^p y_k^p$. This step is performed incrementally by adding $\dot{y}\dot{x}_{k-1}$ for $k > 1$ to the input tape and continuing the computation of the previous cycle.
6. Select the program p with highest rating w_k^p : $p_k^* := \text{argmax}_p w_k^p$.
7. Write $\dot{y}_k := y_k^{p_k^*}$ to the output tape.
8. Receive input \dot{x}_k from the environment.
9. Begin next cycle: $k := k + 1$, goto step 5.

Properties of the p^* algorithm. Let p be any extended chronological (incremental) policy of length $\ell(p) \leq \tilde{l}$ and computation time per cycle $t(p) \leq \tilde{t}$, for which there exists a proof of $\text{VA}(p)$ of length $\leq l_P$. The algorithm p^* , depending on \tilde{l} , \tilde{t} and l_P but not on p , has always higher rating than any such p . The setup time of p^* is $t_{\text{setup}}(p^*) = O(l_P^2 \cdot 2^{l_P})$, and the computation time per cycle is $t_{\text{cycle}}(p^*) = O(2^{\tilde{l}} \cdot \tilde{t})$. Furthermore, for $\tilde{t}, \tilde{l}, l_P \rightarrow \infty$, policy p^* converges to the behavior of the AIXI model.

Roughly speaking, this means that if there exists a computable solution to some AI problem at all, then the explicitly constructed algorithm p^* is such a solution. This claim is quite general, but there are some limitations and open questions regarding the setup time, regarding the necessity that the policies must rate their own output, regarding true but not (efficiently) provable $\text{VA}(p)$, and regarding “inconsistent” policies.

1.8 Discussion

What has been achieved. We suggested an elegant mathematical foundation of artificial intelligence. More specifically, we developed a theory for rational agents acting optimally in any environment. Thereby we touched various scientific areas, including reinforcement learning, algorithmic information

theory, computational complexity theory, probability theory, sequential decision theory, and many more. We presented sequential decision theory in a very general form and unified it with Solomonoff's theory of universal induction, both shown to be optimal in their own domain. The resulting parameter-free AIXI model constitutes an agent for which we gave strong arguments that it behaves optimally in any environment. For restricted environmental classes and Bayes mixtures ξ we showed that $\text{AI}\xi$ is self-optimizing and Pareto optimal. We discussed the choice of the horizon and motivated the use of non-geometric discounting of rewards. We also discussed a number of important problem classes, including sequence prediction, strategic games, function minimization, and supervised learning. All in all, this shows that artificial intelligence *can* be framed by an elegant mathematical theory. Some progress has also been made toward an elegant *computational* theory of intelligence. $\text{AIXI}(tl)$ has optimal order of computation time, apart from a large multiplicative constant, which we could get rid of at the expense of an (unfortunately even larger) additive constant.

Comparison to other approaches. There are many other approaches to AI, too many to mention them all. Among the models that can learn from experience are the “classical” reinforcement learning algorithms like temporal difference learning [SB98], adaptive variants of Levin search [SZW97, Sch04], prediction with expert advice [CB97], market/economy-based reinforcement learning [Bau99, KHS01b], etc. All these models have been implemented and are applicable in limited domains, often with reasonable performance. In contrast, $\text{AIXI}(tl)$ behaves optimally in every (completely general) environment, is data efficient, has generalization capabilities, addresses the exploration versus exploitation problem, etc., but is computationally not feasible without further approximations.

Outlook & open questions. The major theoretical challenge is to derive good non-asymptotic bounds on the value or related quantities for $\text{AI}\xi$ and AIXI, ideally as strong as in the sequence prediction case. The major practical challenge is to scale the $\text{AI}\xi$ model down, e.g. by using more restricted forms of ξ , like the minimum description length principle does for universal induction. The $\text{AIXI}(tl)$ model is a different, very general approach toward a computational model. Unfortunately, it suffers from the same large factor $2^{\bar{l}}$ in computation time as Levin search for inversion problems [Lev73b, Lev84]. On the other hand, Levin search has been implemented and successfully adapted and applied to a variety of problems [Sch97, Sch04, SZW97], and the multiplicative constant can be eliminated as in $M_{p^*}^\epsilon$ or reduced by the Gödel machine [Sch03b]. Inspecting existing approaches suggests that, while AIXI is an elegant mathematical theory that seems to serve all formal needs, computational AI may be messy. For instance, special-purpose algorithms for pre-processing inputs and postprocessing outputs are likely to be necessary in any efficient AI system. Another issue is that of incorporating extra knowledge. In principal there is no need to modify AIXI, since any prior knowledge can

simply be presented as first input x_1 in any format. As long as the algorithm to interpret the data is of size $O(1)$, AIXI will “understand” the data after a few cycles. Another important issue is that of the training process itself. By a training process we mean a sequence of simple-to-complex tasks to solve, with the simpler ones helping in learning the more complex ones. These and many other conceptual, practical, and philosophical issues, including concurrent actions and perceptions, the choice of the I/O spaces, treatment of encrypted information, peculiarities of mortal embodied agents, the free will paradox, the existence of objective probabilities, the Turing test, the existence of efficient and elegant universal theories of intelligence related to Penrose’s non-computable environments and Chaitin’s ‘number of wisdom’ Ω will be addressed later in the book.

1.9 History & References

Introductory textbooks. The book of Hopcroft and Ullman, and in the new revision coauthored by Motwani [HMU01], is a very readable elementary introduction to automata theory, formal languages, and computation theory. The artificial intelligence book [RN95] by Russell and Norvig gives a comprehensive overview over AI approaches in general. For an excellent introduction to algorithmic information theory, Kolmogorov complexity, and Solomonoff induction one should consult the book of Li and Vitányi [LV97], or the book of Calude [Cal02] which focuses more on algorithmic randomness. The reinforcement learning book by Sutton and Barto [SB98] requires no background knowledge, describes the key ideas, open problems, and great applications of this field. A tougher and more rigorous book by Bertsekas and Tsitsiklis on sequential decision theory provides all (convergence) proofs [BT96].

Algorithmic information theory. Kolmogorov [Kol65] suggested to define the information content of an object as the length of the shortest program computing a representation of it. Solomonoff [Sol64] invented the closely related universal prior probability distribution and used it for binary sequence prediction [Sol64, Sol78] and function inversion and minimization [Sol86]. Together with Chaitin [Cha66, Cha75], this was the invention of what is now called algorithmic information theory. For further literature and many applications see [LV97, Cal02]. Other interesting applications can be found in [Cha91, Sch99, VW98, CV03]. Related topics are the weighted majority algorithm invented by Littlestone and Warmuth [LW94], universal forecasting by Vovk [Vov92], Levin search [Lev73b], PAC-learning introduced by Valiant [Val84] and minimum description length [LV92a, Ris89]. Resource-bounded complexity is discussed in [Dal73, Dal77, FMG92, Ko86, PF97], resource-bounded universal probability in [LV91, LV97, Sch02b]. Implementations are rare and mainly due to Schmidhuber [Sch95, WS96, Sch97, SZW97, Sch03a,

Sch04]. Good reviews with a philosophical touch are [LV92b, Sol97]. For an older general review of inductive inference see Angluin [AS83].

Sequential decision theory. The other ingredient in our AIXI model is sequential decision theory. We do not need much more than the maximum expected utility principle and the expectimax algorithm [Mic66, RN95]. The book of von Neumann and Morgenstern [NM44] might be seen as the initiation of game theory, which already contains the expectimax algorithm as a special case. If the true environmental μ is unknown, it needs to be learned with, e.g. the help of reinforcement learning algorithms. Existing reinforcement learning algorithms are [Sam59, BSA83, Sut88, Wat89, WD92, MA93, Tes94, BT96, KLM96, KLC98, WS98, KS98], but they are rather limited in view of AIXI. The literature on reinforcement learning and sequential decision theory is so vast that we refer to the textbooks [SB98, BT96, KV86] for further references.

The author's contributions. Many of the issues addressed in this book can already be found scattered in various reports and publications by the author: The AIXI model was first introduced and discussed in March 2000 in [Hut00] in a 62-page-long report. More succinct descriptions were published in [Hut01d, Hut01e]. The AIXI model has been argued to formally solve a number of problem classes, including sequence prediction, strategic games, function minimization, reinforcement and supervised learning [Hut00]. The generalization $\text{AI}\xi$ has recently been shown to be self-optimizing and Pareto optimal [Hut02b]. The construction of a general fastest algorithm (within a factor of 5) for all well-defined problems [Hut02a] arose from the construction of the time-bounded AIXI^t model [Hut00, Hut01d]. Convergence [Hut03b] and tight [Hut03c] error [Hut01c, Hut01a] and loss [Hut01b, Hut03a] bounds for Solomonoff's universal sequence prediction scheme have been proven. These and other papers are available at <http://www.idsia.ch/~marcus/ai>.

Universal Artificial Intelligence

Sequential Decisions Based on Algorithmic Probability

Hutter, M.

2005, XX, 278 p., Hardcover

ISBN: 978-3-540-22139-5