Towards a Universal Theory of Artificial Intelligence based on Algorithmic Probability and Sequential Decisions

Marcus Hutter

IDSIA, Galleria 2, CH-6928 Manno-Lugano, Switzerland,
marcus@idsia.ch [*]
WWW home page: http://www.idsia.ch/~marcus

Abstract. Decision theory formally solves the problem of rational agents in uncertain worlds if the true environmental probability distribution is known. Solomonoff's theory of universal induction formally solves the problem of sequence prediction for unknown distributions. We unify both theories and give strong arguments that the resulting universal AIξ model behaves optimally in any computable environment. The major drawback of the AIξ model is that it is uncomputable. To overcome this problem, we construct a modified algorithm AIξ^{tl}, which is still superior to any other time t and length l bounded agent. The computation time of AIξ^{tl} is of the order $t\cdot 2^l$.

1 Introduction

The most general framework for Artificial Intelligence is the picture of an *agent* interacting with an environment [RN95]. If the goal is not pre-specified, the agent has to learn by occasional reinforcement feedback [SB98]. If the agent shall be universal, no assumption about the environment may be made, besides that there *exists* some exploitable structure at all. We may ask for the most intelligent way an agent could behave, or, about the optimal way of learning in terms of real world interaction cycles. *Decision theory* formally[1] solves this problem only if the true environmental probability distribution is known (e.g. Blackjack) [Bel57, BT96, SB98]. On the other hand, there is a universal theory for a subset of machine learning, namely, passively predicting unseen data after exposure to training examples. [Sol64, Sol78] formally solved this *induction* problem if the true distribution is unknown, but only if the agent cannot influence the environment (e.g. weather forecasts) [LV97]. Here, we combine both ideas to obtain a universal machine learner for the general case where the learner can actively influence its world. We claim that the resulting *parameterless model AIξ behaves optimally in any computable environment* (e.g. prisoner or auction problems, poker, car driving). To get an *effective solution*, a modification AIξ^{tl}, superior to

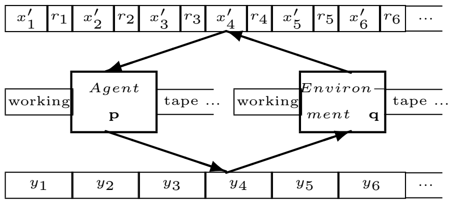
[*] This work was supported by SNF grant 2000-61847.00 to Jürgen Schmidhuber.

[1] With a formal solution we mean a rigorous mathematically definition, uniquely specifying the solution. For problems considered here this always implies the existence of an algorithm which asymptotically converges to the correct solution.

any other time t and length l bounded agent, is constructed. The computation time of AIξ^{tl} per interaction cycle is of the order $t\cdot 2^l$. The computation time is still not practical, but AIξ^{tl} represents the first learning algorithm with generalization capability at all which makes no specific assumptions (Markov property, known transition matrix, observablility, similarity relations, ...) on the structure of the environment. The AIξ and AIξ^{tl} models lead to many new insights into learning agents. The main goal of this work is to derive and discuss both models, and to clarify the meanings of *universal*, *optimal*, *superior*, *etc.* It summarizes a long report and is necessarily succinct. Details can be found in the technical report [Hut00].

2 Rational Agents & Sequential Decisions

Agents in probabilistic environments: A very general framework for intelligent systems is that of rational agents [RN95]. In cycle k, an agent performs *action* $y_k \in Y$ (output word) which results in a *perception* $x_k \in X$ (input word), followed by cycle $k+1$ and so on. If agent and environment are deterministic and computable, the entanglement of both can be modeled by two Turing machines with two common tapes (and some private tapes) containing the action stream $y_1y_2y_3...$ and the perception stream $x_1x_2x_3...$ (The meaning of $x_k \equiv x'_k r_k$ is explained in the next paragraph):



x'_1 r_1 x'_2 r_2 x'_3 r_3 x'_4 r_4 x'_5 r_5 x'_6 r_6 ...
working
Agent p
tape ...
working
Environ-ment q
tape ...
y_1 y_2 y_3 y_4 y_5 y_6 ...

The program p is called the *policy* of the agent interacting with environment q. We write $p(x_{<k}) = y_{1:k}$ to denote the output $y_{1:k} \equiv y_1...y_k$ of the agent p on input $x_{<k} \equiv x_1...x_{k-1}$ and similarly $q(y_{1:k}) = x_{1:k}$ for the environment q. We call Turing machines p and q behaving in this way *chronological*. In the more general case of a *probabilistic environment*, given the history $yx_{<k}y_k \equiv yx_1...yx_{k-1}y_k \equiv y_1x_1...y_{k-1}x_{k-1}y_k$, the probability that the environment leads to perception x_k in cycle k is (by definition) $\mu(yx_{<k}y\underline{x}_k)$. The underlined argument $\underline{x}_k$ in μ is a random variable and the other non-underlined arguments $yx_{<k}y_k$ represent conditions. We call probability distributions like μ *chronological*. Details on the notation can be found in [Hut00].

The AIμ Model: The goal of the agent is to maximize future *rewards*, which are provided by the environment through the inputs x_k. The inputs $x_k \equiv x'_k r_k$ are divided into a regular part x'_k and some (possibly empty or delayed) reward r_k. The μ-expected reward sum of future cycles k to m (called the value) with outputs $y_{k:m} \equiv y^p_{k:m}$ generated by the agent's policy p can be written compactly as

$$V^p_\mu(\dot{y}\dot{x}_{<k}) := \sum_{x_k...x_m} (r_k + ... + r_m)\mu(\dot{y}\dot{x}_{<k}y\underline{x}_{k:m}), \tag{1}$$

where m is the *lifespan* of the agent, and the dots above $\dot{y}\dot{x}_{<k}$ indicate the actual action and perception history. The value with outputs y_i generated by the *ideal agent* which maximizes the expected future rewards is

$$V_\mu^*(\dot{y}\dot{x}_{<k}) := \max_{y_k}\sum_{x_k}...\max_{y_m}\sum_{x_m}(r_k+...+r_m)\mu(\dot{y}\dot{x}_{<k}\underline{yx}_{k:m}), \tag{2}$$

i.e. the best expected reward is obtained by averaging over the x_i and maximizing over the y_i. This has to be done in chronological order to correctly incorporate the dependency of x_i and y_i on the history. The output $\dot{y}_k$, which achieves the maximal value defines *the AIμ model*:

$$\dot{y}_k := \arg\max_{y_k}\sum_{x_k}...\max_{y_m}\sum_{x_m}(r_k+...+r_m)\mu(\dot{y}\dot{x}_{<k}\underline{yx}_{k:m}). \tag{3}$$

The AIμ model is optimal in the sense that no other policy leads to higher μ-expected reward. A detailed derivation and other recursive and functional versions can be found in [Hut00].
Sequential decision theory: Eq. (3) is essentially an Expectimax algorithm/ sequence. One can relate (3) to the Bellman equations [Bel57] of sequential decision theory by identifying complete histories $yx_{<k}$ with states, $\mu(yx_{<k}\underline{yx}_k)$ with the state transition matrix, $V_\mu^*(yx_{<k})$ with the value of history/state $yx_{<k}$, and y_k with the action in cycle k [BT96, RN95, Hut00]. Due to the use of complete histories as state space, the AIμ model neither assumes 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 (3) is much more 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 Y and 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 $\dot{y}_k$ as long as μ is known and computable and X, Y and m are finite.
Reinforcement learning: Things dramatically change if μ is unknown. Reinforcement learning algorithms [KLM96, SB98, BT96] are commonly used in this case to learn the unknown μ. 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, or have non-optimal learning rate. There is no universal and optimal solution to this problem so far. In Section 4 we present a new model and 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 a universal prior ξ, which is shown to converge to μ in a sense.

3 Algorithmic Complexity and Universal Induction

The problem of the unknown environment: We have argued that currently there is no universal and optimal solution to solving reinforcement learning problems. On the other hand, [Sol64] defined a universal scheme of inductive inference, based on Epicurus' principle of multiple explanations, Ockham's razor, and Bayes' rule for conditional probabilities. For an excellent introduction one should consult the book of [LV97]. In the following we outline the theory and the basic results.

Kolmogorov complexity and universal probability: Let us choose some universal prefix Turing machine U with unidirectional binary input and output tapes and a bidirectional working tape. We can then define the (conditional) prefix Kolmogorov complexity [Cha75, Gác74, Kol65, Lev74] as the length l of the shortest program p, for which U outputs the binary string $x=x_{1:n}$ with $x_i\in\{0,1\}$:

$$K(x) \;:=\; \min_p\{l(p):U(p)=x\},$$

and given y

$$K(x|y) \;:=\; \min_p\{l(p):U(p,y)=x\}.$$

The *universal semimeasure* $\hat\xi(\underline{x})$ is defined as the probability that the output of U starts with x when provided with fair coin flips on the input tape [Sol64, Sol78]. It is easy to see that this is equivalent to the formal definition

$$\hat\xi(\underline{x}) \;:=\; \sum_{p\,:\,\exists\omega:U(p)=x\omega} 2^{-l(p)} \tag{4}$$

where the sum is over minimal programs p for which U outputs a string starting with x. U might be non-terminating. As the short programs dominate the sum, $\hat\xi$ is closely related to $K(x)$ as $\hat\xi(\underline{x})=2^{-K(x)+O(K(l(x)))}$. $\hat\xi$ has the important universality property [Sol64] that it dominates every computable probability distribution $\hat\rho$ up to a multiplicative factor depending only on $\hat\rho$ but not on x:

$$\hat\xi(\underline{x}) \;\geq\; 2^{-K(\hat\rho)-O(1)}\cdot\hat\rho(\underline{x}). \tag{5}$$

The Kolmogorov complexity of a function like $\hat\rho$ is defined as the length of the shortest self-delimiting coding of a Turing machine computing this function. $\hat\xi$ itself is *not* a probability distribution[2]. We have $\hat\xi(\underline{x0})+\hat\xi(\underline{x1})<\hat\xi(\underline{x})$ because there are programs p which output just x, neither followed by 0 nor 1. They just stop after printing x or continue forever without any further output. We will call a function $0\leq\hat\rho\leq 1$ with the property $\sum_{x_n}\hat\rho(\underline{x}_{1:n})\leq\hat\rho(\underline{x}_{<n})$ a *semimeasure.* $\hat\xi$ is a semimeasure and (5) actually holds for all enumerable semimeasures $\hat\rho$.

Universal sequence prediction: (Binary) sequence prediction algorithms try to predict the continuation x_n of a given sequence $x_1...x_{n-1}$. In the following we will assume that the sequences are drawn from a probability distribution and that the true probability of a string starting with $x_1...x_n$ is $\hat\mu(\underline{x}_{1:n})$. The probability of x_n given $x_{<n}$ hence is $\hat\mu(x_{<n}\underline{x}_n)$. Usually $\hat\mu$ is unknown and the system can only have some belief $\hat\rho$ about the true distribution $\hat\mu$. Now the universal probability

[2] It is possible to normalize $\hat\xi$ to a probability distribution as has been done in [Sol78, Hut99] by giving up the enumerability of $\hat\xi$. Bounds (6) and (8) hold for both definitions.

$\hat\xi$ comes into play: [Sol78] has proved that the mean squared difference between $\hat\xi$ and $\hat\mu$ is finite for computable $\hat\mu$:

$$\sum_{k=1}^{\infty}\sum_{x_{1:k}}\hat\mu(\underline{x}_{<k})(\hat\xi(x_{<k}\underline{x}_k) - \hat\mu(x_{<k}\underline{x}_k))^2 \;<\; \ln 2\cdot K(\hat\mu) + O(1). \tag{6}$$

A simplified proof can be found in [Hut99]. So the difference between $\hat\xi(x_{<n}\underline{x}_n)$ and $\hat\mu(x_{<n}\underline{x}_n)$ rapidly tends to zero $n\to\infty$ with $\hat\mu$ probability 1 for *any* computable probability distribution $\hat\mu$. The reason for the astonishing property of a single (universal) function to converge to *any* computable probability distribution lies in the fact that the sets of $\hat\mu$-random sequences differ for different $\hat\mu$. Past data $x_{<n}$ are exploited to get a (with $n\to\infty$) improving estimate $\hat\xi(x_{<n}\underline{x}_n)$ of $\hat\mu(x_{<n}\underline{x}_n)$. The learning rule is deeply hidden in the Bayesian mechanism. The universality property (5) is the central ingredient for proving (6).

Error bounds: If we measure prediction quality as the number of correct predictions, the best possible system predicts the x_n with the highest probability. Let SPρ be a probabilistic sequence predictor, predicting x_n with probability $\hat\rho(x_{<n}\underline{x}_n)$. If $\hat\rho$ is only a semimeasure the SPρ system might refuse any output in some cycles n. Further, we define a deterministic sequence predictor SPΘ_ρ predicting the x_n with highest $\hat\rho$ probability. $\Theta_\rho(x_{<n}\underline{x}_n) := 1$ for one x_n with $\hat\rho(x_{<n}\underline{x}_n) \geq \hat\rho(x_{<n}\underline{x}'_n)\,\forall x'_n$ and $\Theta_\rho(x_{<n}\underline{x}_n) := 0$ otherwise. SPΘ_μ is the best prediction scheme when $\hat\mu$ is known. If $\hat\rho(x_{<n}\underline{x}_n)$ converges quickly to $\hat\mu(x_{<n}\underline{x}_n)$ the number of additional prediction errors introduced by using Θ_ρ instead of Θ_μ for prediction should be small in some sense. Let us define the total number of expected erroneous predictions the SPρ system makes for the first n bits:

$$E_{n\rho} \;:=\; \sum_{k=1}^{n}\sum_{x_{1:k}}\hat\mu(\underline{x}_{1:k})(1-\hat\rho(x_{<k}\underline{x}_k)). \tag{7}$$

The SPΘ_μ system is best in the sense that $E_{n\Theta_\mu} \leq E_{n\rho}$ for any $\hat\rho$. In [Hut99] it has been shown that SPΘ_ξ is not much worse

$$E_{n\Theta_\xi} - E_{n\rho} \;\leq\; H + \sqrt{4E_{n\rho}H + H^2} \;=\; O(\sqrt{E_{n\rho}}) \tag{8}$$

$$\text{with}\quad H \;<\; \ln 2\cdot K(\hat\mu) + O(1)$$

and the tightest bound for $\hat\rho = \Theta_\mu$. For finite $E_{\infty\Theta_\mu}$, $E_{\infty\Theta_\xi}$ is finite too. For infinite $E_{\infty\Theta_\mu}$, $E_{n\Theta_\xi}/E_{n\Theta_\mu} \overset{n\to\infty}{\longrightarrow} 1$ with rapid convergence. One can hardly imagine any better prediction algorithm as SPΘ_ξ without extra knowledge about the environment. The values of the $O(1)$ terms in the bounds (6) and (8) depend on the chosen universal Turing machine [LV97]. For real-sized problems (but not for toy problems), $K(\hat\mu) \gg O(1)$. Therefore the bounds, and hence the prediction quality, is only marginally effected by different Turing machine choices. In [Hut01], (6) and (8) have been generalized from binary to arbitrary alphabet and to general loss functions. Apart from computational aspects, which are of course very important, the problem of sequence prediction could be viewed as essentially solved.

4 The Universal AIξ Model

Definition of the AIξ Model: We have developed enough formalism to suggest our universal AIξ model. All we have to do is to suitably generalize the universal semimeasure $\hat{\xi}$ from the last section and to replace the true but unknown probability $\hat{\mu}$ in the AIμ model by this generalized ξ. In what sense this AIξ model is universal and optimal will be discussed thereafter.

We define the generalized universal probability ξ as the $2^{-l(q)}$ weighted sum over all chronological programs (environments) q which output $x_{1:k}$, similar to (4) but with $y_{1:k}$ provided on the "input" tape:

$$\xi(\underline{yx}_{1:k}) \ := \sum_{q:q(y_{1:k})=x_{1:k}} 2^{-l(q)}. \tag{9}$$

Replacing μ by ξ in (3) the *AIξ system* outputs

$$\dot{y}_k := \arg\max_{y_k}\sum_{x_k}...\max_{y_m}\sum_{x_m}(r_k+ ... +r_m)\xi(\dot{y}\dot{x}_{<k}\underline{yx}_{k:m}). \tag{10}$$

in cycle k given the history $\dot{y}\dot{x}_{<k}$.

(Non)parameters of AIξ: The AIξ model and its behaviour is completely defined by (9) and (10). It (slightly) depends on the choice of the universal Turing machine, because $K()$ and $l()$ are defined only up to terms of order 1 [LV97]. The AIξ model also depends on the choice of X and 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 $I\!N$ as word space would be ideal, but whether the maxima (or suprema) exist in this case, has to be shown beforehand. The only non-trivial dependence is on the horizon m. Ideally we would like to chose $m=\infty$, but there are several subtleties to be discussed later, which prevent at least a naive limit $m\to\infty$. So apart from m and unimportant details, the AIξ system is uniquely defined by (10) and (9) without adjustable parameters. It does not depend on any assumption about the environment apart from being generated by some computable (but unknown!) probability distribution, as we will see.

Universality of ξ: It can be shown that ξ defined in (9) is universal and converges to μ analogously to the SP case (5) and (6). The proofs are generalizations from the SP case. The actions y are pure spectators and cause no difficulties in the generalization. This will change when we analyze error/value bounds analogously to (8). The major difference when incorporating y is that in (4), $U(p) = x\omega$ produces strings starting with x, whereas in (9) we can demand q to output exactly n words $x_{1:n}$ as q knows n from the number of input words $y_1...y_n$. ξ dominates all *chronological enumerable semimeasures* [Hut00]

$$\xi(\underline{yx}_{1:n}) \ \geq \ 2^{-K(\hat{\rho})-O(1)}\hat{\rho}(\underline{yx}_{1:n}). \tag{11}$$

ξ is a universal element in the sense of (11) in the set of all enumerable chronological semimeasures. This can be proved even for infinite (countable) alphabet [Hut00].

Convergence of ξ to μ: From (11) one can show

$$\sum_{k=1}^{n}\sum_{x_{1:k}}\mu(y\underline{x}_{<k})\Big(\mu(yx_{<k}y\underline{x}_k)-\xi(yx_{<k}y\underline{x}_k)\Big)^2 \;<\; \ln 2\cdot K(\mu)+O(1) \qquad (12)$$

for computable chronological measures μ. The main complication in generalizing (6) to (12) is the generalization to non-binary alphabet [Hut01]. The y are, again, pure spectators. (12) shows that the μ-expected squared difference of μ and ξ is finite for computable μ. This, in turn, shows that $\xi(yx_{<k}y\underline{x}_k)$ converges to $\mu(yx_{<k}y\underline{x}_k)$ for $k\to\infty$ with μ probability 1. If we take a finite product of ξ-s and use Bayes' rule, we see that also $\xi(yx_{<k}y\underline{x}_{k:k+r})$ converges to $\mu(yx_{<k}y\underline{x}_{k:k+r})$. More generally, in the case of a bounded horizon $h_k\equiv m_k-k+1\leq h_{max}<\infty$, it follows that

$$\xi(yx_{<k}y\underline{x}_{k:m_k}) \stackrel{k\to\infty}{\longrightarrow} \mu(yx_{<k}y\underline{x}_{k:m_k}) \qquad (13)$$

with μ-probability 1. Eq. (13) does not guarantee $\dot y_k^\xi\to\dot y_k^\mu$, since $\dot y_k^{\mu/\xi}$ are discontinuous functions of μ/ξ due to the discontinuous arg max operation in (3/10). This gap is already present in the SPΘ_ρ models, but nevertheless good error bounds could be proved. This gives confidence that the outputs $\dot y_k^\xi$ of the AIξ model (10) could converge to the outputs $\dot y_k^\mu$ of the AIμ model (3), at least for a bounded horizon h_k. The problems with a fixed horizon $m_k=m$ and especially $m\to\infty$ will be discussed later.

Universally optimal AI systems: We want to call an AI model *universal*, if it is μ-independent (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, which can solve or learn significantly faster (in terms of interaction cycles). As the AIξ model is parameterless, ξ converges to μ in the sense of (12,13), the AIμ model is itself optimal, and we expect no other model to converge faster to AIμ by analogy to SP (8),

we expect AIξ to be universally optimal.

This is our main claim. Further support is given in [Hut00] by a detailed analysis of the behaviour of AIξ for various problem classes, including prediction, optimization, games, and supervised learning. The difficulties in obtaining a precise formulation and a general proof, as well as suggestions to overcome them, are analyzed in detail. A first (weak) bound for the passive case is proven.

The choice of the horizon: The only significant arbitrariness in the AIξ model lies in the choice of the lifespan m or the $h_k\equiv m_k-k+1$ if we allow a cycle dependent m. We will not discuss *ad hoc* choices of h_k for specific problems. We are interested in universal choices. The book of [Ber95] thoroughly discusses the mathematical problems regarding infinite horizon systems.

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. Exponential discounting $r_k\rightsquigarrow r_k\cdot\gamma^k$ solves the mathematical problem of $m\to\infty$ but is no real solution, since an effective horizon $h\sim\ln\frac{1}{\gamma}$ has been introduced. The scale invariant discounting $r_k\rightsquigarrow r_k\cdot k^{-\alpha}$ has a dynamic horizon $h\sim k$. This choice has some appeal, as it seems that humans of age k years usually do not plan their lives for more than the next $\sim k$ years. From a practical point of view this model might serve all needs, but from a theoretical point we feel uncomfortable with such a limitation in the horizon from the very beginning. A possible way of taking the limit $m \to \infty$ without discounting and its problems can be found in [Hut00].

Another objection against too large choices of m_k is that $\xi(yx_{<k}\underline{yx}_{k:m_k})$ has been proved to be a good approximation of $\mu(yx_{<k}\underline{yx}_{k:m_k})$ only for $k \gg h_k$, which is never satisfied for $m_k = m \to \infty$. On the other hand it may turn out that the rewards $r_{k'}$ for $k' \gg k$, where ξ may no longer be trusted as a good approximation of μ, are in a sense randomly disturbed with decreasing influence on the choice of $\dot{y}_k$. This claim is supported by the forgetfulness property of ξ (see next paragraph) and can be proved when restricting to factorizable environments [Hut00].

We are not sure whether the choice of m_k is of marginal importance, as long as m_k is chosen sufficiently large and of low complexity, $m_k = 2^{2^{16}}$ for instance, or whether the choice of m_k will turn out to be a central topic for the AIξ 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. We suppose that the limit $m_k \to \infty$ for the AIξ model results in correct behaviour for weakly separable (defined in the next paragraph) μ, and that even the naive limit $m \to \infty$ may exist.

Value bounds and separability concepts: The values V_ρ^* associated with the AIρ systems correspond roughly to the negative error measure $-E_{n\rho}$ of the SPρ systems. In the SP case we were interested in small bounds for the error excess $E_{n\Theta_\xi} - E_{n\rho}$. Unfortunately, simple value bounds for AIξ or any other AI system in terms of V^* analogously to the error bound (8) cannot hold [Hut00]. We even have difficulties in specifying what we can expect to hold for AIξ or any AI system which claims to be universally optimal. In SP, the only important property of μ for proving error 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 classes or consider bounds depending on other properties of μ, rather than on its complexity only. In [Hut00] the difficulties are exhibited by two examples. Several concepts, which might be useful for proving value bounds are introduced and discussed. They include forgetful, relevant, asymptotically learnable, farsighted, uniform, (generalized) Markovian, factorizable and (pseudo) passive μ. They are approximately sorted in the order of decreasing generality and are called *separability concepts*. A first weak bound for passive μ has been proven.

5 Time Bounds and Effectiveness

Non-effectiveness of AIξ: ξ is not a computable but only an enumerable semimeasure. Hence, the output $\dot{y}_k$ of the AIξ model is only asymptotically computable. AIξ yields an algorithm that produces a sequence of trial outputs eventually converging to the correct output $\dot{y}_k$, 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 by time-limited versions [LV97], which is suitable for sequence prediction, has been shown to fail for the AIξ model [Hut00]. This leads to the issues addressed next.

Time bounds and effectiveness: Let $\tilde{p}$ be a policy which calculates an acceptable output within a reasonable time $\tilde{t}$ per 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 AIξ system which 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 AI$\xi^{\tilde{t}\tilde{l}}$ system of computation time $c\cdot\tilde{t}$ per cycle for some constant c. The idea is to run all programs p of length $\leq\tilde{l}:=l(\tilde{p})$ 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 semi-computable and the approximation quality by using computable versions of ξ given a time of order $c\cdot\tilde{t}$ is crude [LV97, Hut00]. On the other hand, we *have* to use a measure which converges to V_ξ^* for $\tilde{t},\tilde{l}\rightarrow\infty$, since we want the AI$\xi^{\tilde{t}\tilde{l}}$ model to converge to the AIξ model in that case.

Valid approximations: A solution satisfying the above conditions is suggested in [Hut00]. 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 AI$\xi^{\tilde{t}\tilde{l}}$ model selects the output $\dot{y}_k=y_k^p$ of the policy p with highest rating w_k^p. 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. In [Hut00] a logical predicate VA(p), called *valid approximation*, is defined, which is true if, and only if, p *always* satisfies $w_k^p\leq V_\xi^p(yx_{<k})$, i.e. never overrates itself. $V_\xi^p(yx_{<k})$ is the ξ 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 AI$\xi^{\tilde{t}\tilde{l}}$ system: In the following, we describe the algorithm p^* underlying the AI$\xi^{\tilde{t}\tilde{l}}$ 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 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 VA($\cdot$) has been formulated. Take those strings which are proofs of VA(p) for some p and keep the corresponding programs p.
2. Eliminate all p of length $>\tilde{l}$.
3. Modify all p in the following way: all output $w_k^p y_k^p$ is temporarily written on an auxiliary tape. If p stops in $\tilde{t}$ steps the internal 'output' is copied to the output tape. If p does not stop after $\tilde{t}$ steps a stop is forced and $w_k^p:=-\infty$ and some arbitrary y_k^p is written on the output tape. Let $\mathcal{P}$ be the set of all those modified programs.
4. Start first cycle: $k:=1$.
5. Run every $p\in\mathcal{P}$ on extended input $\dot{y}\dot{x}_{<k}$, where all outputs are redirected to some auxiliary tape: $p(\dot{y}\dot{x}_{<k})=w_1^p y_1^p ... w_k^p y_k^p$. This step is performed in-

crementally 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: $p_k^* := \arg\max_p w_k^p$.
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 $l(p) \leq \tilde{l}$ and computation time per cycle $t(p) \leq \tilde{t}$, for which there exists a proof of 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 $t_{setup}(p^*) = O(l_P^2 \cdot 2^{l_P})$ and the computation time per cycle is $t_{cycle}(p^*) = O(2^{\tilde{l}} \cdot \tilde{t})$. Furthermore, for $l_P, \tilde{t}, \tilde{l} \to \infty$, policy p^* converges to the behavior of the AIξ 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 VA(p), and regarding "inconsistent" policies [Hut00].

6 Outlook & Discussion

This section contains some discussion and remarks on otherwise unmentioned topics.

Value bounds: Rigorous proofs of value bounds for the AIξ theory are the major theoretical challenge – general ones as well as tighter bounds for special environments μ. Of special importance are suitable (and acceptable) conditions to μ, under which $\dot{y}_k$ and finite value bounds exist for infinite Y, X and m.

Scaling AIξ down: [Hut00] shows for several examples how to integrate problem classes into the AIξ model. Conversely, one can downscale the AIξ model by using more restricted forms of ξ. This could be done in a similar way as the theory of universal induction has been downscaled with many insights to the Minimum Description Length principle [Ris89] or to the domain of finite automata [FMG92]. The AIξ model might similarly serve as a super model, from which specialized models could be derived.

Applications: [Hut00] shows how a number of AI problem classes, including *sequence prediction*, *strategic games*, *function minimization* and *supervised learning* fit into the general AIξ model. All problems are claimed to be formally solved by the AIξ model. The solution is, however, only formal, because the AIξ model is uncomputable or, at best, approximable. First, each problem class is formulated in its natural way (when μ^{problem} is known) and then a formulation within the AIμ model is constructed and their equivalence is proven. Then, the consequences of replacing μ by ξ are considered. The main goal is to understand how the problems are solved by AIξ.

Implementation and approximation: The AI$\xi^{\tilde{t}\tilde{l}}$ model suffers from the same large factor $2^{\tilde{l}}$ in computation time as Levin search for inversion problems

[Lev73]. Nevertheless, Levin search has been implemented and successfully applied to a variety of problems [Sch97, SZW97]. Hence, a direct implementation of the AI$\xi^{\tilde{t}\tilde{l}}$ model may also be successful, at least in toy environments, e.g. prisoner problems. The AI$\xi^{\tilde{t}\tilde{l}}$ algorithm should be regarded only as the first step toward a *computable universal AI model*. Elimination of the factor $2^{\tilde{l}}$ without giving up universality will probably be a very difficult task. One could try to select programs p and prove VA(p) in a more clever way than by mere enumeration. All kinds of ideas like, heuristic search, genetic algorithms, advanced theorem provers, and many more could be incorporated. But now we have a problem.

Computability: We seem to have transferred the AI problem just to a different level. This shift has some advantages (and also some disadvantages) but presents, in no way, a solution. Nevertheless, we want to stress that we have reduced the AI problem to (mere) computational questions. Even the most general other systems the author is aware of, depend on some (more than complexity) assumptions about the environment, or it is far from clear whether they are, indeed, universally optimal. Although computational questions are themselves highly complicated, this reduction is a non-trivial result. A formal theory of something, even if not computable, is often a great step toward solving a problem and has also merits of its own (see previous paragraphs).

Elegance: Many researchers in AI believe that intelligence is something complicated and cannot be condensed into a few formulas. They believe it is more a combining of enough *methods* and much explicit *knowledge* in the right way. From a theoretical point of view, we disagree as the AIξ model is simple and seems to serve all needs. From a practical point of view we agree to the following extent. To reduce the computational burden one should provide special purpose algorithms (*methods*) from the very beginning, probably many of them related to reduce the complexity of the input and output spaces X and Y by appropriate pre/post-processing methods.

Extra knowledge: There is no need to incorporate extra *knowledge* from the very beginning. It can be presented in the first few cycles in *any* format. As long as the algorithm that interprets the data is of size $O(1)$, the AIξ system will "understand" the data after a few cycles (see [Hut00]). If the environment μ is complicated but extra knowledge z makes $K(\mu|z)$ small, one can show that the bound (12) reduces to $\ln 2 \cdot K(\mu|z)$ when $x_1 \equiv z$, i.e. when z is presented in the first cycle. Special purpose algorithms could also be presented in x_1, but it would be cheating to say that no special purpose algorithms have been implemented in AIξ. The boundary between implementation and training is blurred in the AIξ model.

Training: We have not said much about the training process itself, as it is not specific to the AIξ model and has been discussed in literature in various forms and disciplines. A serious discussion would be out of place. To repeat a truism, it is, of course, important to present enough knowledge x'_k and evaluate the system output y_k with r_k in a reasonable way. To maximize the information content in the reward, one should start with simple tasks and give positive reward to approximately the better half of the outputs y_k, for instance.

The big questions: [Hut00] contains a discussion of the "big" questions concerning the mere existence of any computable, fast, and elegant universal theory of intelligence, related to Penrose's non-computable environments, and Chaitin's 'number of wisdom' Ω.

References

[Bel57] R. Bellman. *Dynamic Programming.* Princeton University Press, New Jersey, 1957.

[Ber95] D. P. Bertsekas. *Dynamic Programming and Optimal Control, Vol. (II).* Athena Scientific, Belmont, Massachusetts, 1995.

[BT96] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA, 1996.

[Cha75] G. J. Chaitin. A theory of program size formally identical to information theory. *Journal of the ACM*, 22(3):329–340, 1975.

[FMG92] M. Feder, N. Merhav, and M. Gutman. Universal prediction of individual sequences. *IEEE Transactions on Information Theory*, 38:1258–1270, 1992.

[Gác74] P. Gács. On the symmetry of algorithmic information. *Russian Academy of Sciences Doklady. Mathematics (formerly Soviet Mathematics–Doklady)*, 15:1477–1480, 1974.

[Hut99] M. Hutter. New error bounds for Solomonoff prediction. *Journal of Computer and System Science, in press*, 1999. ftp://ftp.idsia.ch/pub/techrep/IDSIA-11-00.ps.gz.

[Hut00] M. Hutter. A theory of universal artificial intelligence based on algorithmic complexity. Technical report, 62 pages, 2000. http://arxiv.org/abs/cs.AI/0004001.

[Hut01] M. Hutter. General loss bounds for universal sequence prediction. Technical Report, Manno(Lugano), CH, 2001. ftp.idsia.ch/pub/techrep/IDSIA-03-01.ps.gz.

[KLM96] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: a survey. *Journal of AI research*, 4:237–285, 1996.

[Kol65] A. N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems of Information and Transmission*, 1(1):1–7, 1965.

[Lev73] L. A. Levin. Universal sequential search problems. *Problems of Information Transmission*, 9:265–266, 1973.

[Lev74] L. A. Levin. Laws of information conservation (non-growth) and aspects of the foundation of probability theory. *Problems of Information Transmission*, 10:206–210, 1974.

[LV97] M. Li and P. M. B. Vitányi. *An introduction to Kolmogorov complexity and its applications.* Springer, 2nd edition, 1997.

[Ris89] J. J. Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific Publ. Co., 1989.

[RN95] S. J. Russell and P. Norvig. *Artificial Intelligence. A Modern Approach.* Prentice-Hall, Englewood Cliffs, 1995.

[SB98] R. Sutton and A. Barto. *Reinforcement learning: An introduction.* Cambridge, MA, MIT Press, 1998.

[Sch97] J. Schmidhuber. Discovering neural nets with low Kolmogorov complexity and high generalization capability. *Neural Networks*, 10(5):857–873, 1997.

[Sol64] R. J. Solomonoff. A formal theory of inductive inference: Part 1 and 2. *Inform. Control*, 7:1–22, 224–254, 1964.

[Sol78] R. J. Solomonoff. Complexity-based induction systems: comparisons and convergence theorems. *IEEE Trans. Inform. Theory*, IT-24:422–432, 1978.

[SZW97] J. Schmidhuber, J. Zhao, and M. Wiering. Shifting inductive bias with success-story algorithm, adaptive Levin search, and incremental self-improvement. *Machine Learning*, 28:105–130, 1997.