Abstract: A learner's modifiable components are called its policy. An algorithm that modifies the policy is a learning algorithm. If the learning algorithm has modifiable components represented as part of the policy, then we speak of a self-modifying policy (SMP). SMPs can modify the way they modify themselves etc. They are of interest in situations where the initial learning algorithm itself can be improved by experience — this is what we call "learning to learn". How can we force some (stochastic) SMP to trigger better and better self-modifications? The success-story algorithm (SSA) addresses this question in a lifelong reinforcement learning context. During the learner's life-time, SSA is occasionally called at times computed according to SMP itself. SSA uses back-tracking to undo those SMP-generated SMP-modifications that have not been empirically observed to trigger lifelong reward accelerations (measured up until the current SSA call — this evaluates the long-term effects of SMP-modifications setting the stage for later SMP-modifications). SMP-modifications that survive SSA represent a lifelong success history. Until the next SSA call, they build the basis for additional SMP-modifications. Solely by self-modifications our SMP/SSA-based learners solve a complex task in a partially observable environment (POE) whose state space is far bigger than most reported in the POE literature.
12.1 INTRODUCTION

**Bias towards algorithmic regularity.** There is no miraculous universal learning algorithm that will perform well in arbitrary environments [Schmidhuber, 1994; Wolpert, 1996; Schmidhuber, 1997]. Appropriate inductive bias [Utgoff, 1986] is essential. In unknown environments, however, we do not want too specific a bias. How unspecific may it be? The *algorithmic Occam's razor bias* (AORB) assumes that problem solutions are non-random but regular [Solomonoff, 1964; Kolmogorov, 1965; Chaitin, 1969; Li and Vitányi, 1993], without specifying the way in which they are non-random. AORB is highly specific in the sense that it tends to be useless for solving an arbitrary problem from the set of all well-defined problems, almost all of which have irregular solutions [Solomonoff, 1964; Kolmogorov, 1965; Chaitin, 1969; Li and Vitányi, 1993]. But AORB is highly unspecific in the sense that it can help to solve all kinds of "typical", "regular", interesting problems occurring in our atypical, regular universe. This paper, based on [Schmidhuber, 1994], goes beyond our other recent papers exploiting algorithmic regularities for practical machine learning purposes [Schmidhuber, 1995a; Schmidhuber, 1997; Wiering and Schmidhuber, 1996].

**Levin search (LS).** References [Schmidhuber, 1995a; Schmidhuber, 1997] show how a variant of Levin search (LS) [Levin, 1973; Levin, 1984; Solomonoff, 1986; Li and Vitányi, 1993] can find problem solutions with low Kolmogorov complexity and high generalization ability in non-random but otherwise quite general settings. LS is of interest because it has the optimal order of computational complexity for a broad class of search problems. For instance, suppose there is an algorithm that solves certain time-limited optimization problems or inversion problems in $O(f(n))$ steps, where $f$ is a total recursive function and $n$ is a positive integer representing problem size. Then universal LS will solve the same problems in at most $O(f(n))$ steps (although a large constant may be buried in the $O$ notation). Despite this strong result, until recently LS has not received much attention except in purely theoretical studies — see, e.g., [Watanabe, 1992].

**Adaptive LS.** References [Schmidhuber, 1995b; Schmidhuber, 1997; Wiering and Schmidhuber, 1996] note that LS is not necessarily optimal if algorithmic information (e.g., [Li and Vitányi, 1993]) between solutions to successive problems can be exploited to reduce future search costs based on experience. Our adaptive LS extension (ALS) [Schmidhuber, 1995b; Wiering and Schmidhuber, 1996] does use experience to modify LS' underlying probability distribution. ALS can dramatically reduce the search time consumed by successive LS calls in certain regular environments [Wiering and Schmidhuber, 1996].

**Exploiting arbitrary regularities?** This paper goes one step further. Let us call a learner's modifiable components its policy. Policy-modifying algorithms are called learning algorithms. We observe that if our learner wants to exploit arbitrary algorithmic regularities then it must be able to execute arbitrary, problem-specific learning