Multistrategy Learning: When, How and Why

Lorenza Saitta

Università di Torino, Dipartimento di Informatica
Corso Svizzera 185, 10149 Torino (Italy)
saitta@di.unito.it

Abstract

In this paper we describe our experience in designing, implementing and using multistrategy approaches to machine learning. In particular, the system SMART+, WHY and REGAL, each one interpreting "multistrategy" in a different way, will be briefly described, and the lessons learned from their realization commented upon.

1 Introduction

With the growing complexity of the applications of machine learning systems, there has been an increasing need of exploiting the power of multiple (or hybrid) approaches to their development, and several projects have been devoted to this research line. Even though there is no general agreement on what *multistrategy* really means, efforts have been made to find out an adequate definition; for example, Michalski (1991) proposes to classify learning methods according to a set of "knowledge transmutation" processes.

Independently of a precise definition, a wide variety of learning approaches and systems can be considered included in the category; they differ for what they mean for "strategy" or for the implemented type of interaction among strategies. However, they all share the basic fact that at some architectural level different components cooperate to achieve a learning task.

In a learning system, multiple strategies can cooperate according to a variety of configurations and interactions. One of the simplest ways to achieve cooperation is the *Toolbox* idea. In Figure 1 the *Selection* case is described: a number of "strategies" S_i ($1 \le i \le M$) work on (or are related to) the same learning problem (aspect) P. The result R_j is the "best", according to a given criterion, among the results given by each one of the strategies. The choice can be done before using the S_i's (*ex ante* selection), if a-priori information is available, or after using the S_i's (*ex post* selection).

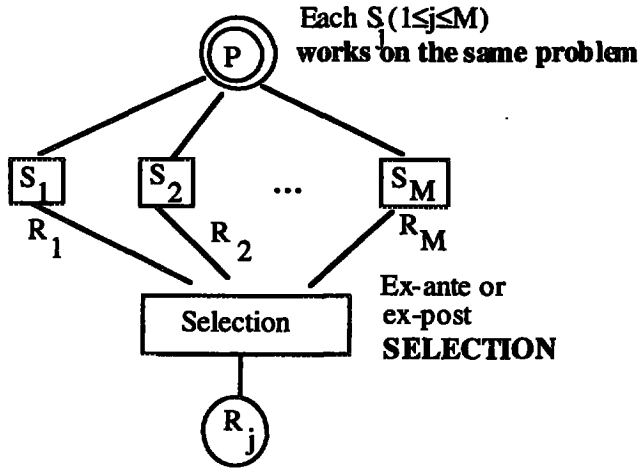

Figure 1 – Toolbox (Selection). Only one among the strategies S_i ($1 \le i \le M$) has to be chosen. This selection can be done before or after the strategies have been used.

A toolbox with selection approach is described in (Kodratoff et al., 1992), where a number of algorithms/systems, oriented to learn rules, concepts, or classes, or combinations of these, are offered; the choice is based, among other parameters, on the type of sought knowledge representation and on the target learning task. In this case, the strategies correspond to algorithms/systems.

A modification of the above schema is the *Toolbox* with *Integration*, represented in Figure 2: the "strategies" S_i ($1 \le i \le M$) again work on (or are related to) the same learning problem (aspect) P. The result R is a function $f(R_1, ..., R_M)$ of the results obtained by all the strategies.

An example of this kind of toolbox integration is given by Drobnic and Gams (1993), who try to combine the results of alternative classification systems to obtain a single final decision.

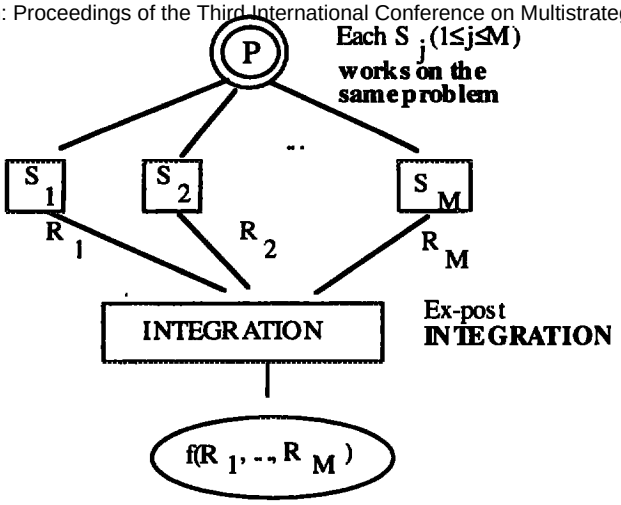

Figure 2 – Toolbox (Integration). The global results is a combination of the results obtained by all the strategies S_i ($1 \le i \le M$).

In the toolbox approach, the strategies S_i work independently, in the sense that each one of them can achieve the common goal by itself, and all of them could be possibly run in parallel. A tighter model of cooperation is *Sequential Cooperation*, represented in Figure 3. According to this schema, each strategy works (possibly iteratively) during a different phase Ph_i of the problem solution, according to a fixed temporal/logical order. The final result is the result provided by the last strategy in the chain.

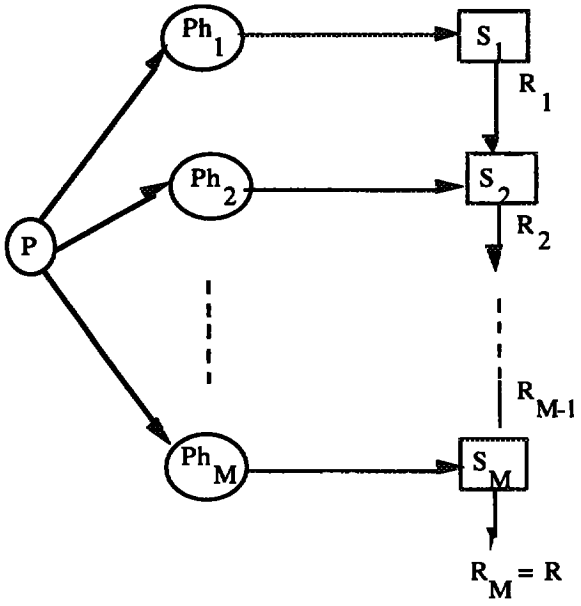

Figure 3 – Sequential Cooperation. The global results is supplied by the last strategy, S_M, in the chain.

Finally, when different phases or aspects of subproblems cannot be isolated, all the strategies work together, as reported in Figure 4, and this cooperation results in a *Tight Integration*.

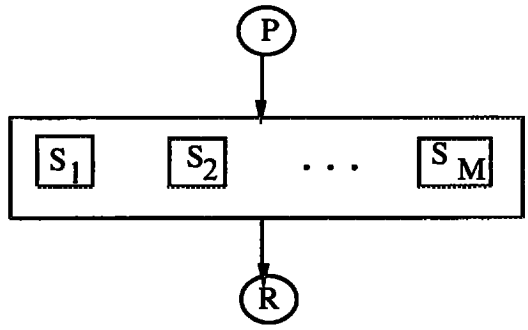

Figure 4 – Tight Integration. The result is provided by the joint work of all the strategies.

Also a mixture of the described schemes is possible. For instance, the *boosting* algorithm AdaBoost (Freud & Schapire, 1995; Dietterich, Kearns & Mansour, 1996) combines the sequential cooperation and toolbox with integration schemes, in the special case in which all the strategies are the same one, consisting of a run of a (weak) learning algorithm; each run is made on a modification of the original training examples distribution, combining then the hypotheses generated at each run through a threshold majority voting.

2 Three Multistrategy Learning Systems

In this section we briefly describe the learning systems SMART+, WHY and REGAL, and discuss the reasons underlying the choice of a multistrategy approach for each one of them.

2.1 The system SMART+

SMART+ (Botta & Giordana, 1993) is a multistrategy learning tool derived from the learning system ML-SMART (Bergadano, Giordana & Saitta, 1991). The system top-down learns concepts, expressed as a set of classification rules in First Order Logic, starting from a set of training examples and a background theory.

Multistrategy occurs in SMART+ at several levels. First of all, different inference schemes are integrated: induction, deduction and abduction. Given a domain theory and a definition of the target concept, deduction is first used as much as possible, in an EBL manner, to set up the frontier of a search tree, whose nodes are then expanded by induction until a halt condition on the quality of the leaves is reached.

Induction adds to the hypotheses under construction subformulas in the specified hypothesis language. During this process, the domain theory may also be used to guide the addition of subformulas, taking into consideration their occurrence in the theory. Let, for instance, the domain theory T contain the two rules describing the stability of a cup:

$$T = \{\text{stable}(x) \leftarrow \text{part-of}(x,y) \wedge \text{flat}(y) \wedge \text{bottom}(y);$$
$$\text{stable}(x) \leftarrow \text{part-of}(x,y) \wedge \text{support}(y) \wedge \text{part-of}(x,w) \wedge \text{body}(w) \wedge \text{above}(w, y)\}$$

and let

$$h = \text{light}(x) \wedge \text{flat}(y) \wedge \text{bottom}(y)$$

be the current hypothesis, to be further specialized. Then, using the theory in an abductive way, we can suppose that the subformula $g = \text{flat}(y) \wedge \text{bottom}(y)$ occurring in h suggests the presence of the concept stable(x), which can be substitute to g in h. If the resulting formula is too general, then stable(x) can be expanded with the second of the two clauses, obtaining thus the new hypothesis:

$$h' = \text{light}(x) \wedge \text{part-of}(x,y) \wedge \text{support}(y) \wedge \text{part-of}(x,w) \wedge \text{body}(w) \wedge \text{above}(w, y).$$

This process may substantially reduce the number of trials of adding new predicates one at a time.

When expanding the search tree, alternative search strategies are available: best-first, deep-first with backtracking or beam search (hill climbing as a special case). These strategies are in alternative, and only one of them can be selected for a given run.

Also the criterion for evaluating a hypothesis can be selected among a set of available ones: information gain, a statistical evaluation of the hypothesis' completeness and consistency, abducibility in the domain theory (i.e., ratio between the number of literals in the hypothesis body and number of them occurring in the theory), and, finally, a combination of the preceding ones.

Considering the kind of data that SMART+ can handle, examples can be described by both categorical and numerical attributes, so as to combine a numerical and symbolic approach to learning. In particular, numerical attributes are discretized by means of fuzzy sets, in order to avoid singularities in the behavior at the discretization points. Finally, the fuzzy sets introduced for each numerical attribute may contain numerical parameters, determining their position and span. The values of these parameters are optimized in two phases: during the hypothesis construction, based on information local to the current node of the search tree, and after a complete hypothesis has been found. At this point, a genetic algorithm further optimizes the parameter values, taking into consideration global information.

By summarizing, SMART+ exploits multistrategy at the level of reasoning strategies, by means of a tight integration scheme. The same kind of integration exists between the numerical and symbolic approaches. On the other hand, SMART+ behaves as a toolbox with selection with respect to the choice of a search strategy and the hypothesis evaluation criterion. Finally, sequential cooperation exists between the symbolic paradigm and the genetic one.

At the beginning, the system ML-SMART only exploited induction with depth-first search strategy, and had a fixed evaluation criterion; moreover, the fuzzy sets for numerical attributes were defined manually by an expert. For the first applications handled, such as recognition of simple images and of speech sounds, the facilities offered by the system were sufficient. As the range and complexity of the applications increased, new requirements were imposed by the end user.

One of the most important was the possibility of explaining the learned knowledge. In order to offer this possibility, the learning system must use a-priori knowledge of the domain, so that inference mechanisms, suitable to exploit this knowledge, have to be incorporated in the system. The availability of domain knowledge, if effectively used, can also provide a strong reduction in the amount of search to be done, focalizing the search towards promising regions.

The need of integration between the numerical and symbolic approaches has its root in the kind of data to be handled in real-world domains, where this co-occurrence is quite frequent. As continuous and categorical features may interact, it is necessary that the same learning algorithm can treat both at the same time, introducing the necessity of some type of discretization. Integration between numerical and symbolic is realized, in SMART+, through a sequential integration type, as the definition of the fuzzy sets (apart from their optimization) precedes the search of the hypothesis space.

Regarding the search strategies and the evaluation criteria, the possibility of offering a choice was motivated by the desire to render the system as flexible as possible, at a low programming expense. Moreover, the multiplicity of possibilities let SMART+ be able to simulate other existing approaches, such as the type of search implemented in decision trees construction.

The reasons of using a genetic algorithm to perform the final optimization are the same that were at the basis of the system REGAL, and will be discussed later.

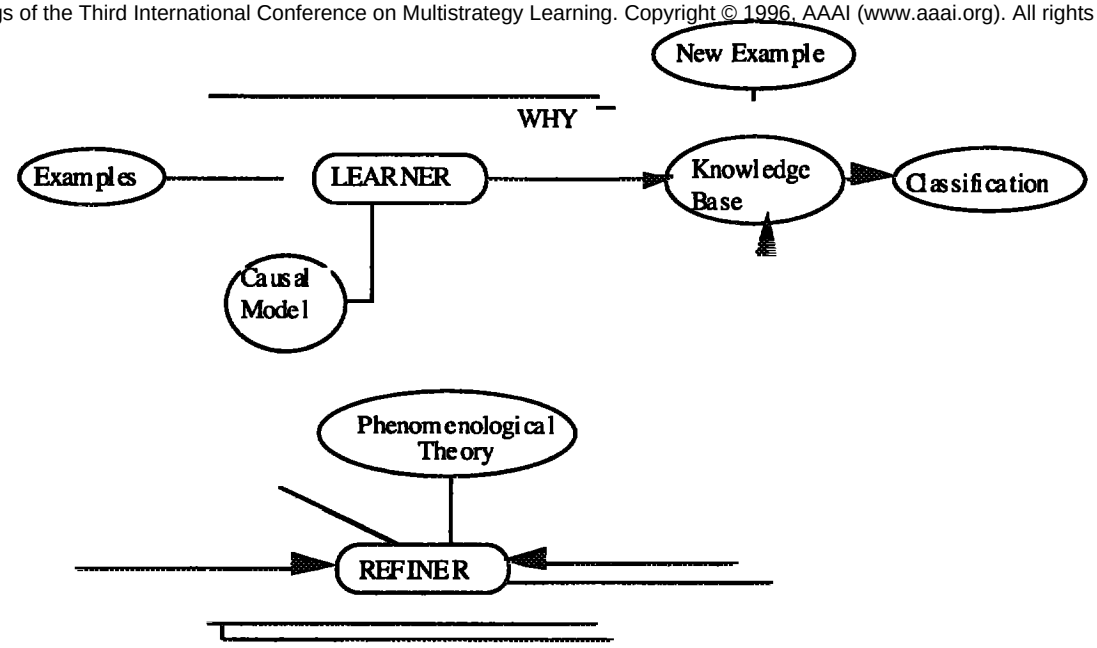

Figure 5 – WHY's architecture. The system is provided with a causal model of the domain and a phenomenological theory and is constituted by two modules: the LEARNER and the REFINER. The LEARNER interprets and explains a set of classified examples, and generates a knowledge base of heuristic rules, allowing classification of new examples to be made upon recognition of specific premises. The REFINER individuates errors and incompleteness both in the knowledge base and in the causal and phenomenological theories.

2.2 The system WHY

WHY is a system that learns and refines a knowledge base for classification tasks, using a *causal model* of the domain, C, a set of examples and a *phenomenological* theory, P, describing concrete manifestations of abstract concepts occurring in C. The system, whose architecture is described in Figure 5, can also be used in a semi-automated way, allowing a direct interaction with a human user (Baroglio, Botta & Saitta, 1994; Saitta, Botta & Neri, 1993).

The representation of all kinds of knowledge used by WHY is based on a First Order Logic language L. The causal model C is represented, at the logical level, as a network. Nodes in the network are either *primary* (ellipses) or *accessory* (rectangles and clouds). Primary nodes correspond to processes or system states and a subset of these nodes contains the *first causes* (shaded nodes). Accessory nodes represent either *constraints* on causal links (rectangles) or *contexts* (clouds), i.e., conditions to be satisfied by the environment. In Figure 6 a causal network describing heat transfer phenomena is represented.

The phenomenological theory P contains structural information, definitions of ontologies, general knowledge and a set of rules describing manifestations associated to abstract concepts. The theory P contains also the links between the causal network C and the classes, which the examples belong to. Examples of pieces of information contained in P are the following ones:

$$\text{CONDUCTION}(x,y) \Leftarrow \text{OBJ}(x) \wedge \text{OBJ}(y) \wedge \text{CONTACT}(x,y)$$
$$\text{HEAT-SOURCE}(x) \Leftarrow \text{fire}(x)$$
$$\text{GOOD-HEAT-COND}(x) \Leftarrow \text{METAL}(x)$$
$$\text{CONTACT}(x,y) \Leftarrow \text{adjacent}(x,y) \vee \text{part-of}(x,y) \vee \text{inside}(x,y)$$
$$\text{METAL}(x) \Leftarrow \text{lead}(x) \vee \text{tin}(x) \vee \text{gold}(x) \vee \text{iron}(x)$$

The target knowledge base K consists of a set of classification rules, derived from the causal network. For instance, from C, the following rule can be learned (x and y denote objects, while T and U denote temperatures):

$$\forall x,y \; [\exists T,U\{\text{OBJ}(x) \wedge \text{OBJ}(y) \wedge \text{diff}(x,y) \wedge$$

$$T \; \text{HERMAL-GAP}(x,y,T,U) \wedge \neg \text{HEAT-SOURCE}(x) \wedge \text{GOOD-HEAT-COND}(y) \wedge \neg \text{HEAT-SOURCE}(y) \wedge \neg\text{CHANGING-STATE}(x) \wedge \neg\neg\text{CHANGING-STATE}(y) \Rightarrow \text{THERMAL-EQUILIBRIUM}(x,y)]$$

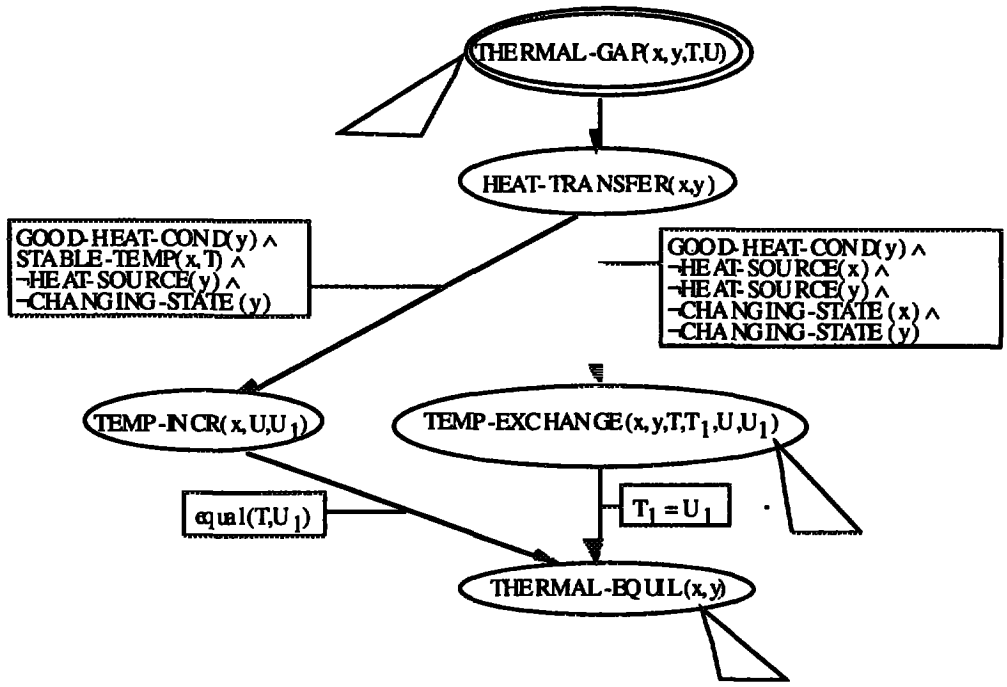

Figure 6 – A causal model C describing heat transfer phenomena.

Four basic reasoning mechanisms are integrated into the system WHY: induction, deduction, abduction and prediction. *Induction* is used when the domain theory is incomplete, and is performed by a call to the system SMART+. Deduction is used to build up a data structure called the *justification forest*, containing links between concrete manifestations and abstract predicates; this forest is a generalization of the explanation tree in EBL. The causal model can be searched either from effects to causes (search for an explanation via *abduction*) or from causes to effects (*prediction* of consequences via deduction).

The system can learn both in one step, from a set of examples, or incrementally, by analyzing one example at a time and updating correspondingly the target knowledge, if necessary. It can also refine a knowledge base, acquired either by itself or supplied by an external source, for instance by an expert. Finally, a semi-automated theory revision is possible, via an interaction with a human expert. WHY was designed as an alternative improvement of ML-SMART, originated by an explicit request by the end user of a complex application of mechanical diagnosis (Giordana et al., 1993). In fact, it seemed important, for both prognosis and repair, not to limit the system answer to the location and classification of faults, but to extend it to include an explanation of why the fault occurred and the identification of its possible causes. In the field of mechanical functioning, qualitative causal models were available. However, in order to effectively exploit these models, the complexity of the system's reasoning scaled up an order of magnitude. Actually, WHY is not a typical learning system, but is rather a general reasoner that extracts from its a-priori knowledge as much information as possible, and tries to drastically reduce the amount of inductive search to be performed.

From the point of view of its structure as a multistrategy system, WHY has the characteristics of a toolbox with integration, with respect to the use of induction and the other reasoning mechanisms. In this case, the "strategies" are whole learning module/systems. Another sense in which WHY is a multistrategy system, using the toolbox with integration scheme, is the cooperation between the learner and the performance module, which exploits both the target heuristic knowledge and the causal network, in case of a difficult classification, not covered by the learned heuristic rules.

The peculiar features of WHY allows tasks, that are difficult to be perform with other learning systems, to be realized. For instance, it was used to investigate the effects of example order presentation on the learning behavior. The presence of a causal model allowed the kind of examples best suited to improve learning in both speed and quality to be predicted, without relying an experiments only based on random order presentations.

Moreover, the "human-like" nature of WHY's learning methods make the system particularly well suited to model simple aspects of human learning. As an example, we are using the system to model conceptual changes in young students learning elementary physics.

2.3 The system REGAL

The last system we consider is REGAL, a learner of First Order Logic concepts based on a genetic algorithm (Neri & Giordana 1995; Neri & Saitta, 1995; Giordana & Neri, 1996). REGAL learns disjunctive concept descriptions by exploiting the theory of niches and species formation (Mahfoud, 1995). Individuals in the population encode partial solutions, i.e. conjunctive formulas, and the whole population is a redundant set of these partial solutions, a subset of which has to be chosen.

Species formation is achieved, in REGAL, by means of two basic mechanisms: on one hand, the selection process is modified by introducing a novel selection operator, called *Universal Suffrage Operator,* and, on the other, the normal population evolution is controlled by a long-term focusing strategy, handled by a special "supervisor" process.

The language for describing the hypotheses is a subset of First Order Logic, namely a Horn clause language, in which terms can be variables or disjunctions of constants (see, for instance, internal disjunction in Induce (Michalski, 1983)), and negation occurs in a restricted form. The set of admissible formulas (hypotheses) is specified by a *Language Template,* which is mapped onto a fixed-length bit string.

The fitness function, used to evaluate individuals, takes into account consistency and simplicity of the hypotheses only, because completeness is taken in care by the Universal Suffrage selection operator.

In REGAL there are four types of crossovers: the *two-point* (De Jong, 1975) and *uniform* (Syswerda, 1989) crossovers, and the *generalizing* and *specializing* crossovers, specifically designed for the task at hand. Finally, a *seeding* operator, reminiscent of the use of a "seed" in Induce (Michalski, 1983), returns a formula covering a given positive instance.

REGAL is a distributed system, working in a computational environment consisting of a network of interconnected subpopulations, exchanging individuals at each generation. The distributed model envisages a SUPERVISOR processor, coordinating a set of *Nodal Genetic Algorithms*, each one searching a subset of the hypothesis space, biased by a different subset of the learning set. The SUPERVISOR realizes an explicit long-term strategy aimed at forming a classification theory (i.e., a complete concept description), with the help of parallel evolving populations. The SUPERVISOR performs two other fundamental tasks: it periodically extracts a classification theory from the global population and, when it finds a satisfactory one, halts the whole system, according to a chosen criterion.

The system REGAL show a tight integration between two learning paradigm: symbolic learning and genetic search. The basic idea motivating the development of this system starts from the consideration that a most critical issue in learning is computational complexity. Instead of reducing the hypothesis space through biases, as is done in most learning systems, we have chosen to leave the search space (and then the representation language) as wide as possible, while increasing, on the contrary, the computational resources dedicated to the search, by exploiting parallelism.

3 Conclusions

In order to face different classes of learning applications, we have developed a number of systems with different characteristics and architectures. The experience gained so far seems to indicate that the complexity of real-world tasks requires a matching complexity in the learning system, in terms of cooperation among components, in such a way that the system results more flexible and adaptable to the application needs. On the other hand, designing more complex systems is a costly process, both in terms of resources required for their realization, and also in terms of speed and ease of use of the learned knowledge. This is especially true when a large amount of a-priori knowledge is to be encoded, so that the improvements in the system outcomes should compensate the greater costs and efforts.

The toolbox approach, however, does not require a large overhead for the integration. On the other hand, it introduces a new problem, namely the problem of the selection. Given a task, there is up to now no guidance (except for a few "rules of thumb" only little less than obvious) to decide what kind of approach or what system is likely to give the "best" performances, according to given criteria.

For multistrategy systems, deciding what components, at what level of the learning architecture should be put together is even more difficult, without a clear understanding of the way the single components are related to the features of the task at hand.

References

Baroglio, C., Botta, M., and Saitta, L. 1994. WHY: A System that Learns from a Causal Model and a Set of Examples. In R. Michalski & G. Tecuci (Eds.), *Machine Learning: A Multistrategy Approach, Vol IV,* 319-348. Los Altos, CA: Morgan Kaufmann.

Bergadano, F., Giordana, A., and Saitta, L. 1991. *Machine Learning: A General Framework and its Applications,* Chichester, UK: Ellis Horwood Ltd..

Botta, M., and Giordana, A. 1993. SMART+: a Multi Strategy Learning Tool. In Proc. 13th Int. Joint Conference on Artificial Intelligence, 937-943. San Mateo, CA: Morgan Kaufmann.

De Jong, K. A. 1975. Analysis of the Behaviour of a Class of Genetic Adaptive Systems, Doctoral Dissertation, Dept. of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI.

Dietterich, T., Kearns, M., and Mansour, Y. 1996. In Proceedings of the 13th Int. Conf. on Machine Learning. San Francisco: Morgan Kaufmann. Forthcoming.

Drobnic, M., and Gams, M. 1993. Multistrategy Learning: An Analytical Approach. In R. Michalski & G. Tecuci (Eds.), Proc. of the Second Multistrategy Learning Workshop, 31-41. Fairfax, VA: George Mason University, Center for Artificial Intelligence.

Freud, Y., and Schapire, R. 1995. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. In Proceedings of Second European Conf. on Computational Learning Theory, 23-37. Springer Verlag.

Giordana, A., and Neri, F. 1996. Search-Intensive Concept Learning. *Evolutionary Computation.* Forthcoming.

Giordana, A., Saitta, L., Bergadano, F., Brancadori, F., and De Marchi D. 1993. ENIGMA: A System that Learns Diagnostic Knowledge. *IEEE Transactions on Knowledge and Data Engineering,* KDE-5: 15-28.

Kodratoff, Y., Sleeman, D., Uszynski, M., Causse, K., and Craw, S. 1992. Building a Machine Learning Toolbox. In L. Steels and B. Lepape (Eds.), *Enhancing the Knowledge-Engineering Process - Contributions from Esprit,* Elsevier Publ. Co.

Mahfoud, S. W. 1995. *Niching Methods for Genetic Algorithms,* Ph.D. Thesis, University of Illinois, Urbana-Champaign, IL.

Michalski, R. 1983. A Theory and Methodology of Inductive Learning. In R. Michalski, J. Carbonell & T. Mitchell (Eds.), *Machine Learning: An AI Approach, Vol. I,* 83-134. Los Altos, CA: Morgan Kaufmann.

Michalski, R. 1991. Inferential Learning Theory as a Basis for Multistrategy Task-Adaptive Learning. In R. Michalski & Y. Kodratoff (Eds.), Proc. of the First Multistrategy Learning Workshop, 3-18. Fairfax, VA: George Mason University, Center for Artificial Intelligence.

Neri, F., and Saitta, L. 1995. Analysis of Genetic Algorithms Evolution under Pure Selection. In Proc. of 6-th Int. Conf. on Genetic Algorithms, 32-41. San Francisco: Morgan Kayfmann.

Neri, F., and Giordana, A. 1995. A Parallel Genetic Algorithm for Concept Learning. In Proc. of 6-th Int. Conf. on Genetic Algorithms, 436-443. San Francisco: Morgan Kaufmann.

Saitta, L., Botta, M., and Neri, F. 1993. Multistrategy Learning and Theory Revision. *Machine Learning,* 11: 153-172.

Syswerda, G. 1989. Uniform Crossover in Genetic Algorithms. In Proc. 3th Int. Conf. on Genetic Algorithms, 2-9. San Mateo, CA: Morgan Kaufmann.