Chapter 4:

Workshop and Panel Overview Papers

Integrated Learning Architectures

E. Plaza[1], A. Aamodt[2], A. Ram[3],
W. van de Velde[4], M. van Someren[5]

[1] *Institut d'Investigació en Intel·ligència Artificial* (CEAB-CSIC), Camí de Santa Bàrbara, 17300 Blanes, Catalunya, Spain. plaza@ceab.es
[2] *University of Throndheim* (Norway). agnar@ifi.unit.no
[3] *Georgia Institute of Technology* (USA). ashwin@cc.gatech.edu
[4] *AI-Lab*, Vrije Universiteit Brussels (Belgium). walter@arti.vub.ac.be
[5] *Universiteit van Amsterdam* (Netherlands). maarten@swi.psy.uva.nl

Abstract. Research in systems where learning is integrated to other components like problem solving, vision, or natural language is becoming an important topic for Machine Learning. Situations where learning methods are embedded or integrated into broader systems offers new theoretical challenges to ML and enlarge the potential range of ML applications. In this position paper we propose the research topic of integrated learning architectures as an initial discussion of the role of learning in intelligent systems. We review the current state of the art and characterise several dimensions along which integrated learning architectures may vary. This paper has been prepared as a position paper with the purpose of providing an initial common ground for discussion in the ECML-93 Workshop on Integrated Learning Architectures. The paper has been edited by E Plaza on the basis of the individual contributions of the authors.

Over the years, AI has divided itself into a number of research areas: planning, learning, vision, knowledge representation etc. Moreover, a multiplicity of learning methods and systems have been developed in the last decade in Machine Learning. There are currently more and more advocates both in ML and AI in general that invite the research community to think about our current situation and re-think our research strategies keeping in mind the long-range goal of a theoretical comprehension and a computational integration of present and future work. In the ML community, a growing trend exists today towards theoretical and implementational integration of the ML methods already developed. We want to bring together in the Integrated Learning Architectures workshop this growing research lines that integrate different ML methods with each other and ML with problem solving. In this paper, we will initiate the discussion of the role of learning in intelligent systems as a key issue on the research agenda on integrated systems.

1. Background

One of the main issues in AI is that of the adaptation of a system to its environment: hardly any system that systematically behaves identically during its interactions with the environment could be considered intelligent. The lack to adequately cope with the adaptation issue has been 'explained' in different ways: lack of flexibility or adaptability, lack of graceful degradation (brittleness), etc. From the beginning of AI, research on learning processes coped with these problems, from adapting to the environment and learning through practice to learning from observation and discovery.

Machine Learning developed into an active field in these last years, and therefore developed its owns goals, techniques, research programmes and paradigmatic achievements. Nevertheless, the necessity of understanding the role of learning processes in intelligent systems remains an open issue.

On the more technical side, building AI systems, both research and commercial, have also suffered from a number of problems, particularly the knowledge acquisition bottleneck and the fragility issue. Machine Learning has been mainly applied to the first problem as an aid to synthesis of new knowledge. Ideally, learning has been viewed, in this paradigm, as a process that from examples (and maybe interactions with some expert users) 'generates' a complete delivery system capable of solving a range of tasks: e. g. use a learning program to build and deliver a rule-based expert system. This approach has a modularity advantage: it is easier (or not much more difficult) in principle to connect ML techniques and AI problem solving techniques when kept both separate. However, an ML-generated delivery system will most likely suffer the same adaptability deficiencies as any other AI system. Yet it is manifest that a lot of flexibility and adaptability can be gained if an AI system, working in a task environment, would be capable of learning from practice, from experience, or from observation, depending on the task at hand (or from all of them). Therefore we have arrived again at the issue of integrating learning with task performance.

The integration of learning into a broader framework is a decisive strategic decision in ML research, from which scientific and technical advances can be gained. Integration issues can establish the nature and goals of learning in the framework of global goals or concerns (e.g. adaptation to environment, routine task speed-up, theory revision, etc.) In fact, it is difficult to conceive how learning can make any sense in complex domains without constraints external to the learning process itself.

Research in ML addresses specific learning tasks in the context of specific problem solvers. In particular the complexity of the full scope of intelligent learning and problem solving is sometimes narrowed down by:

- Assuming a single, uniform task for the performance component. Apart from systems that learn declarative knowledge bases, most systems involve performance tasks that are defined by data and solutions and that have no internal structure.
- Focusing on learning at a single dimension of the performance knowledge (e.g. generalisation, speed up or compression)
- Presenting relevant input data for the learning process in the right form, indicating the role of the data in the learning task. It may not be obvious if and how knowledge is relevant for learning. For example, if we learn to drive a car, it may not be clear if understanding the mechanics of the car is relevant and if so, how this can be used to learn to drive.

Although these simplifications are inevitable in early stages of the research, the limits of this approach are becoming visible. There has also been work in AI generally, and in ML specifically, that is directly concerned with the issues mentioned. All of this deals, in some way or other, with the integration of learning into a more broad architecture. We can arrange them into three main research strategies: cognitive architectures (e.g. SOAR [Newell 90], ACT* [Anderson 83], PI [Holland et al], etc), architectures integrating learning and problem solving (integrated architectures like THEO [Mitchell et al 91], ICARUS [Langley et al], PRODIGY [Carbonell et al], and

case-based reasoning systems like JULIA [Kolodner 87], PROTOS [Porter 90]), and multistrategy learning [MSL-91].

The broader framework in which learning takes place in integrated architectures opens a new spectrum of opportunities for ML research and application. For instance, some issues that were implicit in ML systems need now to be dealt with explicitly. In current research on multistrategy systems, the issue of selecting one learning method among several alternatives is now addressed, and also the criteria that may be used to direct such a selection. In an integrated architecture it is required that the situations that offer opportunities for learning are clearly stated: which kind of situations (failures, success, impasses, etc. and their types) can be exploited by learning, and how? Different options have been explored: impasse-driven learning in Soar, architecture-specific axioms in EBL-Prodigy, etc. These are some of the goals we addressed in the following section.

2. What is an Integrated Learning Architecture?

To qualify as an Integrated Learning Architecture (ILA), a system must be able to learn and to perform at least one problem solving task, and either learning and problem solving must be flexibly integrated in a single control structure, or learning and problem solving must flexibly use the same knowledge structures. The main point here is distinguishing learning embedded in a global system from mere usage of learning methods instrumental to (but external to) build, say, an expert system.

An ILA can be characterised as an architecture for tasks with three different life cycles: simple problem solving, interactive problem solving and learning. These systems can be characterised in terms of their architecture.

Problem Solving	**Data**	**Knowledge**	**Architecture**
Simple PS	presented once & forgotten	stable	stable
Interactive PS	increasing/ being replaced/ forgotten	stable	stable
Learning	increasing/ being replaced/ forgotten	changing	stable

Some tasks are "instantaneous": data are presented and a solution is requested. For other tasks, part of the knowledge remains valid and potentially applicable but other information (data) may change. This is the case in interactive consultation, monitoring systems, etc. Learning systems (in particular incremental learning systems) perform tasks at two levels of scope/time span: (a) the "problem" level (where only *data* change with a new problem) and (b) the "task" or "domain" level: they acquire *knowledge* for performing a particular range of tasks (in addition to type (a) problem solving with changing *data*). Our interest is in systems or methods that perform tasks involving both problem solving and learning and that are "architectures" in the sense that they are described in terms of components which can be configured in different ways. In

particular we want to explore systems performing tasks with both learning and problem solving subtasks.

ILA's define a framework in which a number of new problems can be investigated:

(a) Selecting a learning method

For instance, some issues that were implicit in ML systems need now to be dealt with explicitly. In current research on multistrategy systems, the issue of selecting one learning method among several alternatives is now addressed, and also the criteria that may be used to direct such a selection.

(b) Learning from different types of data

A characteristic of integrated learning architectures is that learning is experiential, i.e. it occurs incrementally through the performance of some reasoning task. Situations that offer opportunities for learning are clearly stated: which of situations (failures, success, impasses, etc. and their types) can be exploited by learning, and how? Different options have been explored: impasse-driven learning in Soar, architecture-specific axioms in EBL- Prodigy, etc.

(c) Selecting data for learning

Planning to learn, detecting opportunities to learn, and selecting learning methods to solve errors of other components of the architecture or improve their performance. The label of "active learning" is used to stress the importance of learning processes in natural intelligent systems, and to emphasise the relevance of learning process in AI systems.

(d) Goal directed learning and problem solving

An ILA has a flexible if not opportunistic learning strategy, in the sense that learning goals that are not immediately satisfiable are remembered so that the reasoner can recognise and use opportunities to pursue them. The learning goals in a ILA can be explicitly represented (e.g. the concepts of SUCCEEDS, FAILS, SOLE-ALTERNATIVE in EBL-Prodigy) or can be implicitly established in the implementation of the architecture.

This raises the following questions:

- What is the goal(s) of learning?
- How learning goals are generated and selected?
- What knowledge of another component does a learning method require in order to be able to learn?
- How do learning and problem solving constrain each other?
- How can they support each other?
- How the results of learning are integrated into the overall architecture?

(e) Which knowledge is needed to perform integrated learning and problem solving?

What knowledge of another component does a learning method require in order to be able to learn? How is that knowledge represented and used?

One approach is to use a *self-model* of the architecture. Self-model(s) is(are) required because of the integration of learning method(s). In general, a learning method has to

have a model of *what are* "successes" and "failures" in the architecture, and of other relevant concepts for learning (e. g. the SOLE-ALTERNATIVE concept in EBL-PRODIGY). These concepts are part of the learning *self-model* of the ILA. Again, the self-model (the definitions of these concepts) can be explicit (as in the architecture axioms of EBL-Prodigy[1]) or can be implicitly established in the implementation of the architecture. These models are method-specific, i.e. they are different for different learning methods (this is called "white-box requirement" of Prodigy in [Carbonell et al], meaning that any ML method has to be able to view and represent what it requires of the system).

Moreover, the learning method needs to be able to effectively inspect part of the structure and behavior (state) of the architecture, and interpret that into its method-specific model. Therefore, *learning is a type of meta-level inference.* A meta-level inference is a kind of inference able to inspect (to have a model of) the object level, infer some new decision, and modify the object-level in such a way that it complies to that decision [Smith 86]. For instance, in EBL-PRODIGY, the architecture has to be able to analyse its behavior, detect situations that involve a failure, select and apply the EBS method, and include the result in its knowledge-base. Meta-level inference in learning has been acknowledged in the literature as the introspective [Ram et al 92] or reflective [Plaza 92] characteristic of embedded learning.

Some examples may clarify this point. Meta-XP in Meta-AQUA [Ram et al 92] record a declarative model of the AQUA system problem solving, determine the blame assignment and selects the adequate method (EBG, index specialisation, etc.) the execution of which transforms the system's knowledge such that its behavior is improved. Another example is the Massive Memory Architecture [Plaza 92], where decisions, successes and failures are declaratively recorded in the system's memory of cases and search control in problem solving is guided (when lacking specific domain knowledge) by analogical transfer of past decisions in similar situations to the current problem.

The meta-level issue is also implicit in the inferential learning theory [Michalski 91]. In ILT learning methods are analysed as higher-level inference patterns the result of which are "knowledge transmutations", i.e. the modification of the system's knowledge as mandated by the inference performed by the learning method.

(f) Theoretical integration: learning and problem solving as inference

Inevitably the issues of integrating learning with different types of problem solving into a coherent whole (an ILA) arises as a necessary element both for ML research specifically, and for the role of learning in AI more broadly. These topics will shape the discussion and understanding of integrating learning, in a principled and comprehensive way, with other kinds of architecture components. Our approach to this is architectural, viewing learning as problem solving, rather than "inference based" as in the work by Michalski [Michalski 91] and inductive logic programming.

---

[1] At least conceptually, architecture axioms are explicit in EBL-Prodigy. However, computationally, the architecture has those axioms implicitly included into the implementation (Alicia Pérez, personal communication).

There are different dimensions along which ILAs may be at variance. One is the typology of opportunities for learning that are established in an architecture. This is part of the method-specific self-model, as for example in EBL-PRODIGY the typology is composed of the following types of learning opportunities SUCCEEDS, FAILS, SOLE-ALTERNATIVE, and GOAL-INTERFERENCE. Another dimension ranges from the fixed attachment of particular learning methods to specific types of situations vs. the dynamic selection of methods in multistrategy learning systems. Prodigy is an example of the fixed attachment of a learning method: only one of the existing methods (EBS, STATIC, derivational analogy, etc.) is used; while Meta-AQUA [Cox et al] selects a learning method according to the type of situation encountered.

A third dimension corresponds to the spontaneous/deliberate occurrence of learning. SOAR and the Massive Memory Architecture are examples of spontaneous learning in the sense that learning taking place automatically and not after an explicit system decision. PRODIGY and Meta-AQUA on the other hand are both examples of deliberate learning because both apply a learning method as an explicit decision resulting from an assessment of the utility of applying a particular learning method to a concrete situation.

3. A Framework for Describing ILAs

In this section we outline a framework for describing integrated learning architectures and propose our research strategy based on that framework. An *integrated architecture* is the computational realisation of a theory in terms of a fixed form, variable contents program that can be instantiated in a systematic and predictable way to achieve a range of systems the behavior of which exhibits a range of aspects of intelligence. Well known integrated architctures are SOAR [Newell 90], ACT* [Anderson 83], PI [Holland et al], etc), THEO [Mitchell et al 91], ICARUS [Langley et al] and PRODIGY [Carbonell et al].

We propose that there are three distinct architectural levels that are meaningful, and useful, to talk about and that correspond roughly to a *why*, *how* and *what* descriptions. These are the knowledge level, the functional (symbol) level and the behavior level.

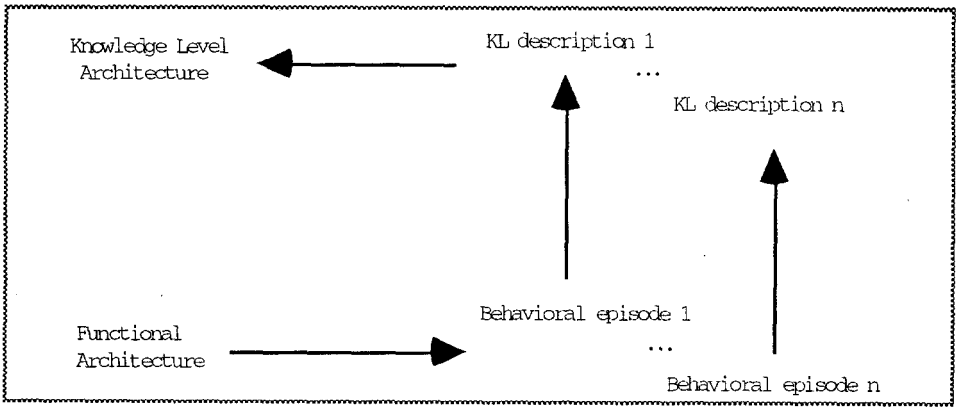
- The behavioral description describes the observable behavior exhibited by a system when it is being applied or executed. This is a *what* model of system behaviour (i.e., a series of episodes of the system's activities) and it is created by an act of *observation.*
- The functional description describes a system in terms of its representational and computational primitives that together constitute the architectural primitives. The functional description is a *how* model of system behaviour and created by an act of *mechanisation.*
- The knowledge level description describes a system in terms of the knowledge of the world and the principles that are applied when using that knowledge (principles of rationality). It is a 'why' model of system behaviour and created by an act of *rationalisation.*

For example a re-implementation of Mycin in SOAR has a behavioral description in terms of consultation episodes, a functional description describing how these consultations are realised as a search in a problem space using certain operators and preference rules, and a knowledge level description that explains the system behavior in

terms of the medical and patient knowledge that it is assumed to embody.

These three perspectives can be used as follows. An integrated architecture is, basically, described at the functional level. That is, it provides one with classes of structures (e.g., state-space, rule, knowledge source, blackboard) and mechanisms (e.g., state-space search, chunking, unification) interconnected and controlled in a fixed way. These structures and mechanisms are the architectural primitives that must be used in order to realise a system behavior. This system behavior, once exhibited, can be described either at the behavioral level or at the knowledge level. Thus, and this is crucial, a single functional architecture is used to implement a range of knowledge level and behavioral descriptions. This makes an architecture a computational theory with "fixed form" but "variable contents" that is targeted towards implementing a range of knowledge level and behavioral descriptions in a controlled and predictable way.

Knowledge level descriptions are, typically, in terms of goals, tasks, methods and models [Steels 90]. Descriptions in these terms capture not only the actual knowledge that is being used, but also the structure that emerges when the knowledge is being put to use in a class of situations. This structure can be called a knowledge level architecture, i.e., a use-specific window on the knowledge. For example, the model of heuristic classification is visible in a pattern of inferences that contribute to abstract, heuristic match and refinement steps in reasoning [Clancey 85]. The model is not visible at the functional level, but it is the recurrent structure for knowledge level descriptions of behavioral episodes exhibited by the system.



We propose to compare architectures by the range of knowledge level architectures that they can realise. Integrated learning architectures will include in their knowledge level architecture, at least one learning goal. Learning goals, tasks and methods are described in exactly the same way as the reasoning aspects. The problem solving aspects of a diagnostic application might be roughly described as follows:

Goal: diagnosis of a car
Task: identify smallest component with functional discrepancy
Method: try shortcut rules, otherwise hierarchical decomposition and testing
Models: heuristic symptom-fault associations, structural model,
expected behavior of components, tests.

The same system could exhibit learning behavior along the following lines:

Goal: reduce number of tests required
Task: acquire shortcut rules from successful episodes
Method: learning through progressive refinement
Models: causal model of component behavior

Implicit in this description is a decision on when to learn (after success), how to learn and what knowledge to use. This is the knowledge level equivalent of the architecture axioms (section 2), and we call them *integration principles*. In other words, an integration principle is a paradigmatic way to interconnect problem solving and learning in order to achieve a learning goal. Note that learning through progressive refinement is not the computational learning technique but an indication of the fact that the system behaves as if it uses such a technique. Whether this is the case or not depends on the functional architecture and as shown in the previous figure, there is no direct mapping from functional architecture to knowledge level architecture.

The above description in terms of goals tasks and methods, without the actual knowledge statements is the knowledge level architecture of the system. In this case it is likely to be fixed. It is however, perfectly possible that the same functional architecture could be used to realise other knowledge level architectures. For example the system might be able to learn from failures as well and maybe include a learning goal of limiting memory overhead which it realises by a task of forgetting infrequently used rules. It may exhibit pursue different learning goals depending on circumstances. In that case it will implement different knowledge level architectures (or a more sophisticated one). For example, the functional architecture of SOAR provides a single learning mechanism (chunking) that can realise a multiplicity of knowledge level learning goals (speedup, increasing goal directedness, smoother interaction, and so forth).

A similar role can be given to the behavior description, though it is typically less central. We propose that classes of behaviors can be described qualitatively to capture behavioral characteristics that make no difference from a rational (knowledge level) point of view but imply different pragmatic constraints. For example the ordering of questions in an interaction episode may be important to determine the practical usability of a system. Behaviors could be characterised as smooth, erratic, aggressive and so forth to describe pragmatic aspects of architectures. These are more prominent for physically behaving systems like robots, and we will not treat this issue further in this paper.

The above scheme is the basis for a research strategy on integrated learning architectures. We propose to analyse functional architectures in terms of the knowledge level architectures that they can realise and, the other way around, to derive ways in which a desired knowledge level architecture can be realised in a given functional architecture. This requires the investigation of knowledge level descriptions of learning goals, tasks and methods in addition to problem solving, and the development of integration principles and architectural axioms that can realise knowledge level architectures in functional architectures.

The framework could be used to describe existing architectures. Here is an initial description of SOAR [Newell 90] and CREEK [Aamodt] architectures. The descriptions are highly incomplete, both in depth and breadth and are intended to serve as an example of characterisations of well-known systems, as well as a proposal to

further discussions and eventually to sum up the architectures presented at the workshop.

3.1 Knowledge Level Description

A knowledge level description of an architecture is a description in terms of the purpose of the architecture and how this purpose is fulfilled by a task decomposition, methods and knowledge. At this level, a system's behaviour in terms of what it intends to do, and what it brings to bear in order to be able to do it, can be explained. The potential and limitations of the system's capabilities can be predicted.

Knowledge-level description of SOAR:

Goal:	Unified cognitive behaviour.
Subtasks:	Solve problem
	Learn during problem solving
Methods:	Problem solving by state-space search, states are existing or created goals.
	Learning by chunking.
Knowledge:	Domain knowledge as productions.
	Strategic knowledge as preferences.

Knowledge-level description of CREEK

Goal:	Problem solving in real world, open domains.
Subtasks:	Solve problem.
	Learn after each problem solving session.
Methods:	Problem solving by combined case-based and generalisation-based reasoning.
	Learning by retaining cases.
Knowledge:	Domain knowledge as a dense semantic net, with cases integrated into it.
	Strategic knowledge as heuristics.

Description items: Goal hierarchy. Tasks assigned to goals. Methods to achieve tasks. Knowledge needed by the methods. A method is applied to a task. This will either lead to an achievement of the task or a decomposition of it into subtasks[2].

3.2 Symbol Level Description

A symbol level description of an architecture is a description in terms of its knowledge representation language, the inference methods of the language, and how these combine into specific reasoning and learning methods. At this level, it can be explained how a system is able to achieve its goals through its methods and its knowledge, by referring

---

[2] See [Steels 90] on knowledge-level descriptions of expert systems and [van de Velde 90] for an application to learning systems.

to the underlying functions that are executed. Predictions can be made about a system's problem solving competence, what it is able to learn, its ability to interact with the external world, etc.

Description items: Knowledge representation language, operations on the representation - in terms of input/output descriptions. A symbol level description, not getting into the actual computational mechanisms.

Symbol-level description of SOAR:

Representation: Problem spaces represents tasks
Productions represent all knowledge
Attribute-value pair is the representational unit

Functions

- probl.solv: Decision cycle: Elaborate, Decide
  Decide: Evaluate preferences, Detect impasse, Create subgoal
- learning: Create chunk after each impasse/subgoaling
- primitive: Select problem space, Select state, Select operator, Apply operator to state.

Symbol-level description of CREEK:

Representation: Single semantic network holds all knowledge
Frames represent all concept types
Frame-slot-facet-value quadruple
(Concept-relation-relation_type-value)
is the representational unit

Functions

- probl.solv: Main cycle: Understand-Generate-Select
  Subcycle for each main function: Activate-Explain-Focus
- learning: Main cycle for case-learning:
  - Extract-Construct-Evaluate-Store
- primitive: Spread activation, Determine context, Retrieve cases, Derive plausible hypotheses, Select best hypothesis, Extract relevant case info, Index case

4. Conclusion

Although there is already some experience in the embedding of learning methods into integrated architectures, most of the crucial issues remain open nowadays. Some of the open issues are general to any computational system that integrates several components, from hybrid representation languages to integrated cognitive architectures. These issues includes the uniform vs. hybrid approaches discussion, and the tight/loose integration spectrum. Uniform approaches like Theo and Soar achieve integration by having all components represented in the same language. Introspective systems like Meta-AQUA and the Massive Memory Architecture achieve integration having a self-model of the system used for learning purposes. Still, other systems like Prodigy have several learning methods that have different models of the problem solving component and thus are separately integrated with the same problem solver but no further integration among them is achieved.

More specifically, embedding learning arises some crucial issues for ML, as we discussed in §2: What knowledge of another component does a learning method require in order to be able to learn? What is(are) the goal(s) of learning? How learning goals are generated and selected in the integrated architecture? How the results of learning are integrated into the overall architecture? Furthermore, the necessity of a comprehensive theory for analysing and comparing different learning and problem solving components arises. One candidate is the Inferential Learning Theory [Michalski 91], another one is using a knowledge-level description like the Components of Expertise [Steels 90] for describing both learning and problem solving components [van de Velde 90], [Plaza 92].

There are also following are some long term research objectives:

- categorisation of learning goals of agents, either as individual [Mitchell 90] or as a group [Brazdil et al 92].
- study of integration principles (architecture axioms) and their applicability conditions. Associated methods for flexible and dynamic (re-)configuration of learning task and methods within problem solving to deal with varying learning goals imposed by the environment.
- techniques for the genuine combination of learning methods, rather than treating them as alternatives to be selected or, alternatively, techniques for the unification of learning methods in a single approach.
- learning beyond domain knowledge, for example of new tasks or methods. Learning about learning (about why, when, what and how to learn).
- learning under resource limitations (anytime learning, memory management, role of forgetting).
- are the processes of learning and problem solving really different van de Velde 90]? Are learning and adaptation really different [Maturana and Varela]?
- integration with physical behaving systems, robots [van de Velde 92] while taking into account recent results from robotics [Maes 90], biology [Maturana and Varela] and epistemology [Clancey 85].

The ultimate goal, of course, is to construct an architecture that embodies the answers to all of these questions. The research strategy outlined above is an approach toward

this goal. It is our feeling that work on isolated learning can not yield significant new insights and that now is the time to try the integration of the results from Machine Learning and other disciplines in a unifying theory and architecture of reasoning, behavior and learning.

Advances in the direction of ILA's will be both of theoretical and practical interest. We expect that it will integrate research on learning and problem solving, increasing our understanding of intelligence. On the other hand it will teach us how to apply learning in the context of intelligent systems, even where these are based on a wide variety of problem solving architectures. This will broaden the range of possible applications of ML techniques.

Acknowledgements

Enric Plaza acknowledges the support of the Massive Memory Project funded by the PRONTIC 90/801 project grant at the IIIA. Ashwin Ram acknowledges the support of the National Science Foundation under grant IRI-9009710 and of the Georgia Institute of Technology. Walter Van de Velde acknowledges the support of the Belgian Ministry of Scientific Research under grant ADIOS (IT/IF/18).

References

[Aamodt 90] Aamodt, A (1990), Knowledge-intensive case-based reasoning and sustained learning. *Proc. ECAI-90*, Stockholm.

[Anderson 83] Anderson, J R (1983), *The Architecture of Cognition.* Harvard University Press: Cambridge.

[Brazdil et al 92] Brazdil, P, et al (1992), Multi-agent learning. In *Proc. EWSL-92.* Speringer Verlag.

[Carbonell et al] Carbonell, J G, Knoblock, C A, Minton, S, (1992), PRODIGY: An integrated architecture for planning and learning. In K VanLehn (Ed), *Architectures for Intelligence*, p. 241-278.

[Clancey 85] Clancey, W (1985), Heuristic classification, *Artificial Intelligence* 27, p. 289-350. North-Holland, Amsterdam.

[Clancey 90] Clancey, W (1990), The frame reference problem in the design of intelligent machines. In Van Lehn and Newell, A (Eds.) *Architectures for Intelligence.* Erlsbaum: Hillsdale, NJ.

[Cox et al] Cox, M T, Ram, A (1991), Using introspective reasoning to select learning strategies. In R Michalski and G Tecuci (Eds.) *Proc. Int. Work. on Multistrategy Learning*, p. 217-230.

[Holland et al] Holland, J H, Holyoak, K J, Nisbett, R E, Thagart, P R (1986), *Induction: Processes of Inference, Learning and Discovery.* The MIT Press: Cambridge, MA.

[Kolodner 87] Janet Kolodner: Extending problem solver capabilities through case-based inference. *Proc. 4th Workshop on Machine Learning*, UC-Irvine, June 22-25 1987. pp 167-178.

[Langley et al] P Langley, K Thompson, W F Iba, J Gennari, J A Allen (in press), An Integrated Cognitive Architecture for Autonomous Agents. In W van de Velde (in press), Editor, *Towards Learning Robots,* MIT Press.

[Maes 90] Maes, P (Ed.) (1990) Special Issue on Designing Autonomous Agents. *Robotics and Autonomous Systems*, **6**(1-2). North-Holland, Amsterdam.

[Maturana and Varela] Maturana, H R , and Varela, F J (1992), *The Tree of Knowledge: the biological roots of human understanding*. Shambala: Boston.

[Michalski 91] Michalski, R S (1991), Inferential learning theory as a basis for multistrategy task-adaptive learning. In R Michalski and G Tecuci (Eds.) *Proc. Int. Work. on Multistrategy Learning*, p. 3-18.

[Mitchell 90] Mitchell, Y (1990), Becoming increasingly reactive. In *Proc. AAAI-90*, p. 1051-1058.

[Mitchell et al 91] Mitchell, T M, Allen, J, Chalasani, P, Cheng, J, Etzioni, O, Ringuette, M, Schlimmer, J C (1991), Theo: a framework for self-improving systems. In K Van Lenhn (Ed.) *Architectures for Intelligence*. Laurence Erlbaum.

[MSL-91] R Michalski and G Tecuci (Eds.) *Proc. Int. Work. on Multistrategy Learning*, p. 217-230. Harpers Ferry, November 7-9, 1991.

[Newell 90] A Newell (1990), *Unified Theories of Cognition*. Cambridge MA: Harvard UniversityPress

[Plaza 92] Plaza, E (1992), Reflection for analogy: Inference-level-reflection in an architetcure for analogical reasoning. *Proc. IMSA'92 Workshop on Reflection and Metalevel Architectures*, Tokyo, November 1992, p. 166-171.

[Porter 90] Bruce Porter, Ray Bareiss, Robert Holte: Concept learning and heuristic classification in weak theory domains. Artificial Intelligence, vol. 45, no. 1-2, September 1990. pp 229-263.

[Ram et al 92] Ram, A, Cox, M T, Narayanan, S. (1992), An architecture for integrated introspective learning. *Proc. ML'92 Workshop on Computational Architectures for Machine Learning and Knowledge Acquisition.*

[Smith 86] Smith, B C, (1986), Varieties of self-reference. In *Theoretical Aspects of Reasoning about Knowledge*, p. 19-43, Morgan Kaufmann, Los Altos, CA.

[Steels 90] L Steels (1990), The Components of Expertise, *AI Magazine*, August 1990.

[van de Velde 90] W --van de Velde, W (1990), Reasoning, Behavior and learning: A knowledge-level perspective . *Proc. of Cognitiva 90*, pp. 451-463. Madrid 20-23 Nov. 1990.

[van de Velde 92] van de Velde, W (Ed.) (1992), Toward Learning Robots.Special Issue of *Robotics and Autonomous Systems*, **8**(1-2). North-Holland, Amsterdam.