On Philosophical Incidences of Logic Programming

Luís Moniz Pereira*
Centro de Inteligência Artificial - CENTRIA, Universidade Nova de Lisboa,
2829-516 Caparica, Portugal

Abstract

We address the import of AI for philosophical logic and examine the requirements posed by knowledge representation and reasoning issues which AI has addressed, most especially through its Logic Programming field, to more dynamic forms of logic, in order to deal with knowledge in flux. In the process, we consider its encroachments on the philosophy of knowledge.

1 Introduction

We begin by addressing the general impingement of artificial intelligence (AI) on philosophical logic, by examining the requirements posed on logic by knowledge representation and reasoning issues which AI has addressed. We then outline some of AI's contributions, most especially via the field of Logic Programming (LP), to more dynamic forms of logic, in order to deal with knowledge in flux, namely: incomplete and contradictory information; hypotheses making through abduction; argumentation; diagnosis and debugging; updating; and learning. Along the way we delve into implications for the philosophy of knowledge.

2 Evolution and Reasoning

Evolution has provided humans with symbolic thought, and symbolic language communication abilities. Objective common knowledge requires thought to follow abstract, content independent rules of reasoning and argumentation, which must not be entirely subjective, on pain of making precise communication and collective rational endeavour impossible. Such rules have become ingrained in human thought, and hold an enormous joint survival value [7, 8, 15].

In human cognitive evolution, both mimetic knowledge (such as that inherent in reality simulating maps and models), and imitation knowledge (such as that present in ritual observation, or in artifact reproduction), were essential first steps towards socially situated, joint rule following behaviour, required by, say, hunting plans. Subsequently, throughout human cultures, abstract rule following social games emerged.

Game rules encapsulate concrete situation defining patterns, and concrete situation-action-situation causal sequencing, which mirrors causality-obeying physical reality. From games, further abstraction ensued, and there finally emerged the notions of situation-defining concepts, of general rules of thought and their chaining, and of legitimate argument and counter-argument moves. Together they compose a cognitive meta-game [10].

The pervasiveness of informal logic for capturing knowledge and for reasoning, a veritable *lingua franca* across human languages and cultures, rests on its ability to actually foster rational understanding and common objectivity. Crucially, objective knowledge evolution dynamics itself, whether individual or plural, follows ratiocination patterns and laws. Moreover, and more recently, the very same rules of reasoning can and are employed to reason about reasoning. Though some reasoning methods are well known, some are still unconscious but, like the rules of grammar, can be discovered through research. What is more, new reasoning methods can and have been invented and perfected throughout human history. Examples of these are transfinite induction, *reductio ad absurdum* (proof by contradiction), recursion, abduction, and contradiction removal, to name but a few.

New purported reasoning methods may be disputed, just like any specific train of reasoning can. But reasoning can only be disputed by further reasoning, if any consensus is to be found! (cf.[14]). Some argue that scientific and philosophical discussion is necessarily tantamount to a culture sensitive, and culturally relative, persuasive informal *ad hoc* argumentation, allied to anything goes rhetoric [9]. They ignore that argumentation is just another a form of reasoning which has itself been made the subject of logical formalization, and are oblivious to that rhetoric may be fine for preachers, but is not conducive to the two-sided communication required to reach common agreement in the all rigorous scientific praxis that lead to cumulative knowledge.

3 Logic and the Computer

Logic, we have seen, has arisen as the content independent formulation of the Laws of Thought. Predicates express whatever conceptual relations we may wish,

*lmp@di.fct.unl.pt http://centria.fct.unl.pt/~lmp

about the external and internal worlds. These predicate building blocks can then be combined into formulas, by means of quantifiers and logical connectives, and can be manipulated according to inference rules characterizing divers forms of reasoning. Logic is capable of articulating an extensional behaviouristic view of predicates, seen as black boxes or input/output relations, simultaneously with an intensional view of predicates, seen as functionally decomposable into other predicates, like black boxes wired inside black boxes.

Since the language of logic is symbolic and its rules content-independent, its workings can be specified by general, abstract, rule following procedures. These, in turn, can be programmed on the computer. Through their mechanization, logical theories and reasoning forms attain, for the first time, an *in vitro* existence, an ability and availability for repeatable execution, independently of any human mind hardware support.

The two views above, extensional and intensional, are reconciled in the computer by insisting that the declarative semantics, the "what" is to be computed, be equivalent to the procedural computational semantics, the "how" it is computed. This is the cornerstone of the *LP* paradigm.

4 Philosophy and AI

Philosophy has been a cradle for rational thought throughout human intellectual history. Its study of language fostered the refinement of logic as an abstraction of natural language and as a tool of linguistic inquiry. The investigations on the foundations of mathematics too assigned to logic a meta-instrumental rôle that would deploy it as the general purpose symbol manipulating tool *par excellence*. Computer science as well has adopted logic as its general foundational tool, while *AI* has made viable the proposition of turning it into a *bona fide* computer programming language. At the same time, *AI* has developed logic beyond the confines of monotonic cumulativity, typical of the precise, complete, endurable, condensed, and closed mathematical domains, in order to open it up to the non-monotonic real world domain of imprecise, incomplete, contradictory, arguable, revisable, distributed, and evolving knowledge. In short, *AI* has added dynamics to erstwhile statics.

In *AI*, as with any science, common logic has an important rôle in it. Indeed, any science, implicitly or explicitly, presses logic and reasoning into its service, to build up arguments, counter-arguments, and to settle disputes. But a greater rôle can be expected of *AI* in logic! First, *AI* means to mechanize logic, it being such a core tool for so many rational activities. Second, *AI* intends to make explicit, and well-defined, the unconscious logic we use, and put it to objective test by automating it on the computer. Third, *AI* employs logic as a generic language of communication, of knowledge and of procedures, between humans and computers, and among computers themselves. Fourth, in *AI*, even when procedures and devices are not implemented using logic directly, logic can play the rôle of a precise specification language for their requirements, and of a formalism with which to study their semantical properties. Fifth, *AI* has contributed significantly to the phrasing and examination of the problem of identifying limits to symbolic reasoning embodied in computers, and whether these limits apply to humans as well. Sixth and final rôle, *AI* has helped researchers explore new reasoning issues and methods, and to combine disparate reasoning modalities into a uniform unified framework, so as to deal with incomplete, imprecise, contradictory, and changing information. To this rôle we now turn.

5 A Logical Tool of AI

Classical logic has been developed to study well-defined, consistent, and unchanging mathematical objects. It thereby acquired a static character. *AI* needs to deal with knowledge in flux, and less than perfect conditions, by means of more dynamic forms of logic. Much of this has been the focus of research in *LP*, a field of *AI* which uses logic directly as a programming language, and provides specific implementation methods and efficient working systems to do so. *LP* is moreover much used as a staple implementation vehicle for other *AI* approaches to logic.

5.1 Logic Programming

There is a close connection between logical implication and physical causality. After all, logic permits us to model the goings on in the world. This is even more so in the case of *LP*, where all true conclusions must be supported, "caused", by some premises, and where implication is unidirectional, i.e. not contrapositive: "causes" do not run backwards. Below, so-called Horn clause notation is used to express this directionality. The ability for *LP* to model actions by means of updates is brought out in the section on updating.

To produce a result or conclusion C we need a conjunction of enough positive conditions P to sustain it, conjoined by ',' to the absence or negation of a conjunction $\neg N$ of all the negative conditions that would prevent it, given P:

$$C \leftarrow P, \neg N$$

If there is more than one way to obtain C, then we have several rules of this form:

$$C \leftarrow P_i, \neg N_i$$

If our information about each rule, and about the whole

set of rules for C, is complete, then we would have:

$$C \longleftrightarrow \bigvee_i P_i, \neg N_i$$

But what if we don't? First, we might not have enough information about each $\neg N_i$. We might conceivably know about each required P_i, without at least one of which C can never be achieved. But regarding the $\neg N_i$, i.e. all the conditions which, if present, would prevent us from concluding C in the presence of each P_i, that's more difficult.

5.2 Open worlds

Too many things can go wrong in an open non-mathematical world, some of which we don't even suspect. For all we know, some bomb might go off that destroys the whole physical setup we're trying to model logically (at a safe distance). Must we model that too? In the real world, any setting is too complex already for us to define it exhaustively each time. We have to allow for unforeseen exceptions to occur, based on new incoming information. Thus, instead of having to ensure or prove that some condition N_i is not present, we can assume it is not, with the proviso that we are prepared to accept subsequent information to the contrary. I.e. we may assume a more general rule than warranted, but must henceforth be prepared to deal with arising exceptions. Take for example, with the obvious reading, this piece of knowledge, again in Horn clause form:

$$faithful(H, K) \leftarrow married(H, K), \neg lover(H, L)$$

We don't normally have explicit information about who is the lover of whom, though that kind of information may arrive unexpectedly, especially of presidents

$$faithful(H, K) \leftarrow married(H, K), not\ lover(H, L)$$

This is expressible via so-called default default negation *not* P, which may be read as "P is not provable". Ie. we no longer require proof that $\neg lover(H, L)$ for any L, given some H, but assume so unless we can establish $lover(H, L)$ for some L given H. In other words, if I have no evidence to conclude $lover(H, L)$ for some L given H, I can assume it false for all L given H.

5.3 Defeasible Assumptions

Default negation was introduced by AI researchers via LP, and may be read, interchangeably, as "P is not provable", or "the falsity of P is assumable", or "the falsity of P is abducible", or "there is no evidence for P", or "there is no argument for P". Default negation allows us to deal with lack of information, a common situation in the real world. It introduces non-monotonicity into knowledge representation. If later we're informed about a pair of lovers, we must then go back on our previous conclusion about whatever faithfulness we had presumed. Indeed, conclusions might not be solid because the rules leading to them may be defeasible. Legal texts, regulations, and courts employ this form of negation abundantly, as they perforce deal with open worlds, though it had not before been formalized in logic till AI too discovered its need of it.

5.4 Closed World Assumption

Mark that *not* should grant positive and negative information equal standing. That is, we should be able to write:

$$\neg faithful(H, K) \leftarrow married(H, K), not\ \neg lover(H, L)$$

to model instead a world where people are unfaithful by default or custom, and where it is required to explicitly prove that someone does not take any lover before we conclude that person not unfaithful. The issue arises because often, particularly in data bases and knowledge bases, the (CWA) in enforced. That is, the CWA obtains when we presume the database or knowledge base to have all the pertinent positive information, so that only what is explicitly stated, or explicitly derivable from it, is true; otherwise, it is presumed false. But how could it, in general, have all the pertinent information in a changing world?

Information is normally expressed positively, by dint of mental and linguistic economics. So, by CWA, the absent, non-explicitly obtainable information is usually the negation of positive information. Which means, when no information is available about lovers, that $\neg lover(H, L)$ is true by CWA, whereas $lover(H, L)$ is not. This asymmetry is undesirable, inasmuch predicate names are purely conventional and we could, as justifiably, have come up with positive predicate $non-lover(H, L)$ and its negation $\neg non-lover(H, L)$ to model our knowledge.

5.5 Explicit Negation

In addition, when we have no factual or derivable information, either positive or negative, we'd like to be able to say that both are false epistemically, i.e. from the "knowledge we possess" point of view. Accordingly, the excluded middle postulate is unacceptable because some predication or other and its negation may be false simultaneously. The CWA and the symmetry and epistemological requisites, can be reconciled by reading '$\neg$' above not as classical negation, which complies with the excluded middle postulate stating that any predication is either true or false, but as yet a new form of negation, dubbed in LP "explicit negation"[5] (which ignores the excluded middle provision). Now we can state the CWA for just those predicates P or $\neg Q$ we

wish, simply by writing:

$$\neg P \leftarrow not\ P \ \text{ or } \ Q \leftarrow not\ \neg Q$$

Alternatively, the use of a *not* P or of a *not* $\neg Q$ assumption may be made at just those predicate occurrences requiring them, like we did before.

5.6 Revising Assumptions

Let us next examine the need for revising assumptions and for introducing a third truth-value, call it "undefined", into our framework. When we combine the viewpoints of the two above worlds we become confused:

$$faithful(H,K) \leftarrow married(H,K), not\ \neg lover(H,L)$$

$$\neg faithful(H,K) \leftarrow married(H,K), not\ lover(H,L)$$

Assuming $married(H,K)$ for some H and K, it now appears that both $faithful(H,K)$ and $\neg faithful(H,K)$ are contradictorily true. Because we have no evidence for $lover(H,L)$ nor $\neg lover(H,L)$, there simply is no information about them, we make two assumptions about their falsity. But when an assumption leads to contradiction one should retract it. Yes, it is the venerable principle of *reductio ad absurdum, or* " reasoning by contradiction". In our case, two assumptions led to the contradiction. Which shall we retract? They are on equal footing!

5.7 Undefinedness

Given no other eventually preferential information, we retract both because we cannot justly decide between them. That is, we assume neither $lover(H,L)$ nor $\neg lover(H,L)$ false. Since neither is provably true also, we make each *undefined*. I.e. we introduce a third truth-value to better characterize this lack of information about some lovers' situation, thereby making $faithful(H,K)$ and $\neg faithful(H,K)$ undefined too. This imposition of undefinedness can be achieved simply, by adding to our knowledge:

$$\neg lover(H,L) \leftarrow not\ lover(H,L)$$

$$lover(H,L) \leftarrow not\ \neg lover(H,L)$$

Given no other information, we cannot prove either of $lover(H,L)$ or $\neg lover(H,L)$ true, or false. Any attempt to do so runs into a self-referential circle involving default negation. However, once we do hypothesize the one true the other perforce becomes false, and vice-versa. These two possible situations are not thus ruled out. But the safest, skeptical, third option is to take no side in this marital dispute, and abstain from believing either. Indeed, the well-founded semantics of logic programs (WFS) assigns to the literals in the above two clauses the truth value *undefined* in its knowledge skeptical well-founded model, and allows also for the other two, non truth-minimal, more credulous models. But why can we not simply add:

$$lover(H,L) \ \vee \neg lover(H,L) \ \ ?$$

Because it will not do. We would still not be able to prove either $lover(H,L)$ or $\neg lover(H,L)$ definitely true, and the contradiction would ensue all the same!

When dealing with non-provability one really needs a third truth-value to express our epistemic inability to come up with information. Even assuming the world is ontologically 2-valued, our access to information coded about it might be 3-valued in general. Besides, the world may very well not be 2-valued, or, in any case, not capturable in a 2-valued way. Can we say of some particle/wave that it is either here or not here? Is it really either here or not here, though we cannot say it? The inherent dispute apparently cannot be settled experimentally. In any case, we are in wont of a third logical value for other reasons. As we build up our real world imperfect knowledge base, we may very well create, unwittingly and unawares, circular dependencies as above. For example, the Legislator may well enact conflicting, circular, laws. Still, we want to be able to carry on reasoning, whether or not such circularities legitimately express what they model. And when they do not, we want to detect and function with them rather than throwing away the baby with the bath water.

But undefinedness may also be legitimately pressed into service to express common knowledge, in its variously credulous and skeptical readings. Why, even yesterday I read in a book [6] of the old sailor proverb that "He weathers the storms he cannot avoid, and avoids the storms he cannot weather", i.e.:

$$weather \leftarrow storm,\ not\ avoid$$

$$avoid \leftarrow storm,\ not\ weather$$

What will happen to an experienced sailor in a storm? Will he weather it or avoid it? Three outcomes are possible, including being undecided about the outcome. For us, it means we must weather circular knowledge, not avoid it in the calm waters of the artificial mathematical paradises...

5.8 Expressiveness

The introduction of this 3-valued semantics into LP is an important innovation of AI, with a number of consequences. Namely, we have seen, it allows us to be skeptical and suppose no more than is warranted. Another innovation, to recapitulate, is that the negation $\neg$ used above, called explicit negation, is 3-valued too, and so differs from classical negation. Explicit negation, we've seen, does not conform to the principle of

excluded middle. Predications and their negations can be both false ("the chair is angry", "the chair is not angry"). In a knowledge base, we may definitely not be able to prove either something or its (explicit) negation, so that both become false by default. Additionally, the introduction of explicit negation, and in particular its combination with default negation, provides for more expressive knowledge representation. Consider for instance the injunction

$$\neg cross \leftarrow train$$

versus the alternative injunction

$$\neg cross \leftarrow not\ \neg train$$

The latter is clearly a safer option. According to it you must actually prove that a train is not coming before $\neg cross$ becomes false, whereas with the first option $\neg cross$ becomes false solely on the grounds that you fail to prove a train is coming!

Yet another innovation is that the implicational arrow $\leftarrow$, when combined with default or explicit negation, is not material implication. Contrapositives are not countenanced. Consider:

$$\begin{array}{rcl} A & \leftarrow & B \\ & & \neg A \end{array}$$

$\neg B$ does not follow, for there is no presumption of the contrapositive $\neg B \leftarrow \neg A$. Whenever contrapositives are desired they must be explicitly added. Instead, $\leftarrow$ should be seen as expressing an inference rule, which can be used procedurally "bottom-up" to conclude A given B, or "top-down", like an invoked procedure, to try and prove the body B in order to prove the head A. Facts are just procedures with empty body. This inferential reading of clauses as rules turns $\leftarrow$ into a directional operator, which makes it rather appropriate to model not only inference but causality as well. Indeed, the semantics of logic programs emphasizes exactly this, with their insistence on admitting only minimal models. Minimal models are those for which every positive or explicitly negated literal *true* in the model is supported on some rule whose head or conclusion is the literal, and whose body or set of premises is in turn supported. Thus facts, since they have an empty body, are automatically supported. Additionally, any positive or any explicitly negated literal is *false* just in case all rules for it have a false body. Consequently, if there are no rules whatsoever for a literal it is automatically false. All other positive or explicitly negated literals are *undefined* (this can happen only if they are involved in unresolved self-referential loops through default negation). Finally, a default negated literal, *not* P, is *true/ false/ undefined* just in case P is, respectively, *false/ true/ undefined*.

5.9 Theory Diagnosis

Many examples exist of the use of default negation, explicit negation, contradicition removal, and 3-valuedness, made possible by research in *LP*, namely in the areas of abduction, argumentation, belief revision, knowledge updating, learning, and diagnosis[3]. Let us illustrate the latter, i.e. the diagnosis or debugging of a generic knowledge base.

Suppose we have the following rules, expressing a logic program that allows us to compute predicate C on the basis of predicates P and N:

$$C(X) \leftarrow P(X), \neg N(X)$$

$$\begin{array}{ll} P(a) & \neg N(a) \\ P(b) & \\ P(c) & \neg N(c) \end{array}$$

Let us allow for the chance that the rule for C is conceivably wrong. i.e. that it may have exceptions, by writing it thus:

$$C(X) \leftarrow P(X), \neg N(X),\ not\ exception(C(X))$$

If nothing else is stated about the *exception* predicate, the conclusions of the logic program remain unchanged, so the program may contain these rules from the start, for any predicate we wish. If we next learn that $C(a)$ is false, we can salvage the rule for C, since it is no longer valid for all $X = a$, simply by adding to our knowledge:

$$exception(C(a))$$

The rule for C will continue to work for the remaining cases, though more exceptions might subsequently be accumulated.

This method takes care of rules that are unsound, i.e. produce wrong results. What about the other problematic case, that of incompleteness, i.e. when results that would be correct are missing? Suppose we learn that $C(b)$ should true, but which our rules rule out? Well, it is enough to introduce for every set of rules for a predicate a catch-all rule. In this case:

$$\begin{array}{rcl} C(X) & \leftarrow & missing(C(X)) \\ P(X) & \leftarrow & missing(P(X)) \\ \neg N(X) & \leftarrow & missing(\neg N(X)) \end{array}$$

If nothing else is stated about the *missing* predicate, the conclusions of the logic program remain unchanged, so the program may contain these rules from the start. Now, to make $C(b)$ true, it now suffices now to abduce, or adopt, one of two hypotheses: that $missing(C(b))$ or that $missing(\neg N(b))$ is true.

This method takes care of rules that are incomplete, i.e. fail to produce results. We have now achieved generality. By making the two predicates, $exception(\_)$ and $missing(\_)$, abducible, we can diagnose and debug a knowledge base expressed in logic.

Abduction is a well-known reasoning process by which one can adopt hypotheses to the effect that some predicate instances (or their negations) are true in order to prove some conclusion. It's reasoning from goals to requirements, and has been extensively studied in the *LP* context [11]. Of course, the assumptions or hypotheses so adopted may later prove erroneous, if and when they lead to contradiction. We have then to make provision for the application of a contradiction removal process based on revising assumptions, possibly adopting of some other assumptions instead. This too has been extensively studied in *AI* and in the *LP* setting, and as a result automated reasoning systems have been made available which do that work for us [1].

5.10 Updating

Last but not least, work in *LP* has concerned itself with the updating of a knowledge base by another one. This notion of knowledge updating, as opposed to that of simple fact updating, opens up another dimension to the dynamics of logic, in contradistinction to the statics of the logic of old. Given an existing knowledge base, containing facts and rules, and some new knowledge with facts and rules as well, possibly contradicting the previous knowledge when added to it, what is the resulting updated knowledge base? Can this process be iterated?

Most of the work conducted so far in the field of *LP* has focused on representing *static* knowledge, i.e., knowledge that does not evolve with time. This is a serious drawback when dealing with *dynamic knowledge bases* in which not only the *extensional* part (the set of facts) changes dynamically but so does the *intensional* part (the set of rules). Recent work has shown how this can be achieved with generality [2, 4]. Application areas in point are when the two knowledge bases consist in pieces of legislation, or regulations, or safety procedures, or rules of robot conduct.

The common intuition behind the update of a model of the world has been based on what is referred to as the commonsense law of inertia, i.e. things do not change unless they are expressly made to do so. Suppose, for example, that we have a model in which "sunshine" is true and "rain" is false; if later we receive the information that the sun is no longer shining, we conclude that "sunshine" is false, due to the update, and that "rain" is still false by inertia.

Suppose now that our vision of the world is described instead by a logic program and we want to update it. Is updating each of its models enough? Is all the information borne by a logic program contained within its set of models? The answer to both these questions is negative. It is not sufficient to just consider the truth values of literals figuring in the heads of its rules, because the truth value of their rule bodies may also be affected by the updates of other literals. In other words, it suggests that the *principle of inertia* should be applied not just to the individual literals in an interpretation but rather to the *entire rules of the knowledge base*.

This approach was first adopted in [12], where the authors presented a program transformation which, given an initial program and an update program, produces an updated program obeying rule inertia. The stance of the dynamic *LP* approach is precisely that of ensuring that any added update rules are indeed in force, and that previous program rules are still in force (by inertia) only in so far as possible, i.e. they are kept for as long as they do not conflict with any newly added ones.

Updating inevitably raises issues about revising and preferring, and some work is emerging on the articulation of these distinct but highly complementary aspects. And learning is usefully seen as successive approximate change, as opposed to exact change, and combining the results of learning by multiple agents, multiple strategies, or multiple data sets, inevitably poses problems within the province of updating. Finally, goal directed belief revision can be fruitfully construed as abductive updating. Thus, not only do the aforementioned topics combine naturally together – and so require precise, formal, means and tools to do so – , but their combination results in turn in a nascent complex architectural basis and component for *LP* rational agents, which can update one another and common, structured, updatable blackboard agents. It can be surmised, consequently, that the fostering of this meshing of topics within the *LP* community is all of opportune, seeding, and fruitful. Indeed, application areas such as software development, multi-strategy learning, abductive belief revision, model-based diagnosis, agent architecture, and others, are being successfully pursued while employing precisely this outlook.

6 A Dynamical Logic Framework

It is not too difficult to imagine how a combined process of rule generation, of systematic diagnosis, and of rule revision by updating, can be used to achieve automated theory learning, in an integrated way, within the uniform setting of *LP*.

To initiate the learning, one starts with some fixed, already acquired, background knowledge in rule form, i.e. a theory, and with a rule generator to add to it new purported knowledge, in order to explain abductively known observations, whether positive or negative, in the form of facts and explicitly negated facts. The goal being to generate rules that define a positive concept as well as its negated concept, so that they cover all known observation instances. This automatic generation of new rules is subjected to a pre-defined bias, i.e. only some rule forms, and predicates comprising them, are allowed in the generation process. Newly generated rules may contradict one another, on some of the ob-

servation instances, and so they must be subjected to a diagnosis, to identify alternative possible minimal revisions. The whole process will be iterated on the basis of new incoming knowledge, or by knowledge confrontation of among differently evolved automated theories, with distinct backgrounds, biases, rule generators, diagnosers, revisors, preferences, planners, observations, and updating procedures, comprising a rational agent.

The use of *LP* for the overall endeavour is justified on the basis of it providing a rigorous single encompassing theoretical basis for the aforesaid topics, as well as an implementation vehicle for parallel and distributed processing. Additionally, *LP* provides a formal high level flexible instrument for the rigorous specification and experimentation with computational designs, making it extremely useful for prototyping, even when other, possibly lower level, target implementation languages are envisaged.

7 Concluding Remarks

I hope to have convinced you that *AI*, most especially through *LP*, will continue to accomplish a good deal in identifying, formalizing, and implementing the Laws of Thought. Most notably, *AI* has taken on the challenge of opening up logic to the dynamics of knowledge in flux. And in so doing, it has been progressively meeting our expectations and requirements.

The *LP* paradigm provides a well-defined, general, integrative, encompassing, and rigorous framework for systematically studying computation, be it syntax, semantics, procedures, or attending implementations, environments, tools, and standards. *LP* approaches problems, and provides solutions, at a sufficient level of abstraction so that they generalize from problem domain to problem domain. This is afforded by the nature of its very foundation in logic, both in substance and method, and constitutes one of its major assets.

Indeed, computational reasoning abilities such as assuming by default, abducing, revising beliefs, removing contradictions, updating, belief revision, learning, constraint handling, etc. etc., by dint of their generality and abstract characterization, once developed can readily be adopted by, and integrated into, distinct topical application areas.

References

[1] J.Alferes, C.Damásio, L.Pereira, "A logic programming system for non-monotonic reasoning". J. Automated Reasoning, 14:93-147, 1995.

[2] J.Alferes, J.Leite, L.Pereira, H.Przymusinska, T.Przymusinski, "Dynamic Updates of Non-Monotonic Knowledge Bases", The Journal of Logic Programming 45(1-3): 43-70, September/October 2000

[3] J.Alferes, L.Pereira, "Reasoning with logic programming", LNAI 1111, Springer, 1996.

[4] J.Alferes, L.Pereira, "Logic programming updating: a guided approach". In F.Sadri et al.(eds),"Computational Logic: From Logic Programming into the Future - Essays in honour of Robert Kowalski", Springer 2002.

J. J. Alferes and L. M. Pereira, Logic Programming Updating - a guided approach , In A.Kakas and F.Sadri (eds), Computational Logic: From Logic Programming into the Future - Essays in honour of Robert Kowalski, Springer, 2002.

[5] J.Alferes, L.Pereira, T.Przymusinski, "'Classical' Negation in Nonmonotonic Reasoning and Logic Programming", J. Automated Reasoning, 20:107-142, 1998.

[6] J.Barth, "On with the story", page 55, Back Bay Books, Little Brown and Company, 1996

[7] T.Deacon, "The Symbolic Species", W.W. Norton, 1997.

[8] M.Donald, "Origins of the Human Mind", Harvard U.Press, 1991.

[9] P.Gross, N.Levitt, "Higher Superstition", The Johns Hopkins U. Press, 1994.

[10] J.Holland, "Emergence", Addison-Wesley, 1998.

[11] A.Kakas, R.Kowalski, F.Toni, "Abductive Logic Programming", J.Logic and Computation, 2:719-770, 1993.

[12] J.Leite, L.Pereira, "Generalizing updates: from models to programs", Procs. ILPS'97 Workshop on Logic Programming and Knowledge Representation, LPKR'97, Port Jefferson 1997.

[13] J.Leite, J.Alferes, L.Pereira, "Multi-dimensional Dynamic Knowledge Representation", Procs. 6th Int. Conf. on Logic Programming and Nonmonotonic Reasoning (LPNMR'01), T. Eiter et al.(eds), 365-378, Springer, LNAI 2173, Springer 2001.

[14] T.Nagel, "The Last Word", Oxford U. Press, 1997.

[15] S.Pinker, "How the mind works", W.W.Norton, 1997.