A Case for Paraconsistent Logic as a Foundation of Future Information Systems

Hendrik Decker

Instituto Tecnológico de Informática
Ciudad Politécnica de la Innovación, Valencia, Spain
email: hendrik@iti.es

Abstract. Logic links philosophy with computer science and is the acknowledged foundation of information systems. Since the large scale proliferation of the internet and the world wide web, however, a rush of new technologies is avalanching, in many cases without much consideration of a solid foundation that would be up to par with the rigor of the traditional logic fundament. Philosophy may help to question established foundations, especially in times of technological breakthroughs that seem to override such foundations. In particular, the intolerance associated with the consistency requirements of classical logic begs question of its legitimacy, in the face of ubiquitous inconsistency in virtually all information systems of sizable extent. Based on that, we propose to overcome classical logic foundations by adopting paraconsistency as a foundational concept for future information systems engineering (ISE).

1 Introduction

Most computational issues can be described and investigated by logic-based formalisms, because logic has a capacity of abstraction that enables the extrication of conceptual issues from idiosyncratic details of their implementation. In this sense, the role of logic for ISE can somehow be likened to the role of philosophy for the humanities (except that the connection between logic and technology seems to be more palpable and less vague than the connection between philosophy and, say, theology or jurisprudence or linguistics).

Since the beginnings of computer science, logic has assumed a foundational role for the technology and the applications of computer science. Logic has linked the formal and conceptual foundations of computer science with its practical applications. Logic has been an acknowledged key pillar in such seemingly different areas as that of hardware, programming languages, software engineering and, perhaps most of all, databases and ISE.

Since the 1990ies, however, the role of logic seems to be on the retreat. At least in part, this can be blamed on several interrelated technological developments: the proliferation of internet and world wide web, in tandem with the synergetic convergence of computing, telecommunication and information retrieval. These, together with the ever-increasing power and miniaturization of hardware components, keep on spurring the imagination of developers and marketing strategies,

yielding an avalanche of projects, prototypes, products and services that move forward seemingly without too much preoccupation for a solid foundation.

In times of foundational crises, philosophy may help to ask questions about otherwise undisputed certainties on which a successful discipline has relied in the past. Such questioning may help to overcome inherent limitations of current foundations that might constitute part of the crisis.

Already more than half a century ago, the concept of paraconsistency arose, when several philosophers questioned the validity of classical logic (abbr. CL) with regard to the *ex contradictione quodlibet* rule, *ECQ*. Roughly, ECQ says that, in the presence of inconsistency, anything, and thus nothing useful at all, can be inferred as an answer to any given query or problem. Thus, CL completely fails to capture that information systems (abbr. ISs) which contain some inconsistent pieces of information may very well produce reasonably useful answers. Hence, it comes as no big surprise that, since about the mid 1990ies, a resurgence of interest in paraconsistent logic for computer science can be observed.

Informally speaking, paraconsistency is is the paradigm of reasoning in the presence of inconsistency [26]. CL intolerantly invalidates any useful reasoning if there is any inconsistency, no matter how irrelevant it may be. However, inconsistencies, as unpleasant and dangerous as they can be, are ubiquitous in information systems. For novel technology which often is not sufficiently mature before being launched on the market, the risk of inconsistencies is even higher. Hence, a thoroughly revised inconsistency-tolerant logic fundament is needed for databases and ISE, also because many future applications (e.g., the self-organizing cognitive evolution of networked information systems, involving negotiation, argumentation, diagnosis, learning, etc) are likely to deal directly with inconsistencies as inherent constituents of real-life situations.

In this paper, we argue for adopting paraconsistency as a new, more realistic inconsistency-resilient logic foundation of ISE. We think that PHISE should be an appropriate forum for such an argument, since it is of speculative nature and based more on philosophical and historical than on purely technical insights.

In section 2, we survey the history and importance of the mutual relationships between philosophy, logic, computer science and ISE. In section 3, we argue why a new paraconsistent foundation of ISE is deemed necessary. In section 4, we propose a procedural definition of paraconsistency which is then shown to correspond to the more usual declarative definition of paraconsistency. We also show that *datalog*, extended with integrity constraints, is paraconsistent in the defined sense, and opine that it could serve as a point of departure for a new paradigm of paraconsistent foundations for future information systems.

2 Historical Perspectives

In 2.1, we point out the historical roots of logic in philosophy, and its importance for ISE. In 2.2, we recapitulate the foundation crisis of mathematics and logic which preceded and gave rise to the emergence of paraconsistency. In 2.3 (and then again in 4.2), we trace some of the early history of paraconsistent logic.

2.1 Philosophy, Logic, Computer Science and ISE

Western philosophy, deductive mathematics and logic have accrued hand in hand in ancient Greece. Philosophy has often sought (though never really reached) the evidence, certainty and simplicity of axioms and methods with which mathematics is obtaining provably correct results. The closest thing to that has been achieved in the discipline of logic. Logic originated as a stylization of rational speech, thought and deductive inference. Its two most outstanding contributors have been the philosophers Aristotle and Frege. After Frege, logic split up into two branches, mathematical and philosophical logic, but both essentially continue to recur on the same foundations.

While logic was battling with the consequences of the crisis as addressed in 2.2, the field of computer science arose, essentially out of combined efforts in applied mathematics, engineering and logic, as recalled subsequently.

While solving basic logic problems of computability, Turing conceived the machine named after him, in 1936 [31]. In 1937, Stibitz used boolean logic for designing and implementing switching circuits in electromechanical arithmetic calculators [29], and Shannon showed that switching circuits can be described by boolean logic [28]. In the 1940ies, Zuse, who had anticipated Stibitz' and Shannon's insights already in 1936 [40], introduced boolean and predicate calculus-based operations of logical inference in his Plankalkül programming language [41]. Proof-theoretic logic, rooted in Gentzen's work in 1935 [10], and model-theoretic logic, as founded by Tarski in 1933 [30] (both partly anticipated by the philosopher Bolzano almost a hundred years earlier [4]), continue to fuel research and development in the fields of databases, ISE and programming languages. As for today's research frontiers, Wadler's work is just one example which highlights the relevance of those early achievements [39].

2.2 The Crisis Preceding Paraconsistency

The philosopher Kuhn [20] has pointed out that, in science, philosophy is mostly asked for in times of crises, when the evolution of "normal" science (which does not doubt underlying basic paradigms) has encountered its limits, beyond which its paradigms become paradoxical or contradictory. Then, more fundamental thought is asked for, questioning hitherto undisputed paradigms and reflecting on possibly revolutionary ways to overcome basic conflicts.

One such crisis occurred around the turn of the 19th to the 20th century, when Frege's set-theoretic axiomatization of mathematics turned out to be inconsistent (cf. [13]). This raised doubts about the logic foundations in general, and initiated many attempts to overcome the problem by new axiomatizations of set theory, mathematics and alternative logic foundations.

Although mathematics as such is not of concern in this paper, the logics developed on the grounds of its foundations crisis are, with regard to their treatment of contradictions and inconsistencies. In particular, we are going to pay attention to the renouncement of certain logical formalisms and laws, by which inconsistencies and unwanted consequences of ECQ are supposed to be avoided.

The three most prominent approaches to provide a contradiction-free logic foundation of mathematics are Russell & Whitehead's logicism, Hilbert's formalism and Brouwer's intuitionism (cf. [19]). While logicism and formalism have remained within the Aristotle-Boole-Frege-Russell-Herbrand-Gentzen lineage of CL, intuitionism questioned some of the most fundamental axioms of logic itself, in particular the law of excluded middle *TND* (*tertium non datur*). Roughly, it says that each statement must either be true or false; no other truth value or modality conceivably exists which would allow a statement to neither hold nor not hold, but to have that third value or modality.

2.3 The Rise of Paraconsistent Logic

A radical departure from formerly undoubted CL foundations had been started off by Jaśkowski [14] and, later but independently, da Costa [8]. They established the field of paraconsistent logic, in which, by various proposals of different non-classical calculi, some fundamental laws of CL are depreciated, such as the law of double negation and even the arguably most fundamental of all logical axioms, the law of contradiction *LoC*. In general, the idea is to ban as few axioms as necessary for invalidating ECQ, while keeping as many axioms as possible, in order to continue to be able to draw sufficiently many valid conclusions.

Arguably the most important precursors of paraconsistent logic were Vasiliev and Łukasiewicz, both philosophers in their own right. Łukasiewicz recalled that Aristotle had discussed several different modalities of LoC, by which certain shadings of clarity and certainty of this most fundamental of principles became apparent [22]. Analogous to the non-euklidian geometry of Lobachevsky and Poincaré (also philosopher, btw, who abandoned the axiom of parallels, which formed the basis for Einstein's revolution of crisis-ridden physics), Vasiliev and Łukasiewicz proposed to study non-aristotelian logics.

The philosopher Pierce, who also doubted TND, had suggested that before them, but did not follow through his initial thoughts (cf. [3]). In Vasiliev's "imaginary logic", LoC was no longer postulated. Also Łukasiewicz thought of abandoning LoC. However, after re-discovering the findings of the medieval philosopher Pseudo-Scotus (to whom is attributed the discovery of ECQ as a consequence of Aristotelian logic [19], and influenced by Kolmogorov (the first to axiomatize Brouwer's intuitionism [18], Łukasiewicz ceased to question LoC and abandoned TND, instead. Akin to Pierce's suggestions, and going beyond the mere cancellation of TND, he investigated multi-valued semantics, from which most multi-valued semantics of databases and information systems stem.

Even though TND does not hold in multi-valued logics, ECQ usually does, i.e., contradiction still entails trivialization (cf. [34]). So, it is not surprising that most variants of paraconsistent logic either radically weaken or completely renounce LoC, while the possibly more doubtful TND is adhered to in most cases. (For more details, cf. [6].) However, in section 4, we are going to interpret *datalog* and some extensions thereof as a paraconsistent formalism which refrains from using TND but does not depart from LoC, while ECQ never applies.

3 Why Paraconsistent Logic for Information Systems ?

Our objective is to look for a more realistic logical foundation for future ISs and ISE. Paraconsistency is the logical paradigm that we deem to be more adequate for this purpose than CL, as argued for in 3.1. The survey in 3.2 further motivates the need for a unifying paraconsistent foundation of different approaches that are all, in one way or another, related to ISE.

3.1 Motivation

As already said, CL takes an absolutely unforgiving stand against inconsistency, although the latter, for better or worse, frequently occurs in human knowledge and artefacts and ultimately cannot be avoided. Thus, CL completely fails to capture that, in practice, inconsistent ISs may very well produce reasonable results. As opposed to that, paraconsistent logic constrains CL, so that useful operations and results are not invalidated ("trivialized") by inconsistency, but can continue to be produced, interpreted, reasoned about and further processed.

Working in an inconsistent context is not just a technicality. On the contrary, intransigent intolerance against inconsistency has been the cornerstone of CL since its incipation, more than two millennia ago, until the present. Applying the ECQ rule has disastrous deductive consequences, yielding a total failure to support a rational understanding of what is going on in an inconsistent context. Moreover, future ISE is supposed to be able to deal directly with inconsistency as an inherent constituent of some application domain, e.g., when there are multiple but contradictory sources of information, or when the user disagrees with given answers and wants to argue with the IS about conflicting points of view.

Hence, there is a clear need to recur on a solid foundation which, unlike CL, can provide a well-grounded understanding, and is able to make dependable predictions, of what is going on in the presence of inconsistency. Paraconsistent logic thus appears to be the most auspicious candidate of such a foundation of ISE, i.e., for describing, explaining, predicting, empowering and operating information systems, just as CL formalisms have been used to describe, explain, predict, empower and operate traditional ISs. Based on that, future ISE is expected to result in more robust and sustainable products than the trendy short-lived items that often come out of new ideas which lack a profound basis.

3.2 Survey of the State-of-the-Art

Computational approaches and applications dealing, in one way or another, with inconsistency, can roughly be classified as belonging to one of three communities: Computational Logic (CoL), Artificial Intelligence (AI) and Philosophical Logic (PhL) (while problems of Mathematical Logic with regard to inconsistencies of set theory, infinitesimal analysis and the like are not of interest in this paper). With regard to ISE, CoL has brought forward Datalog [1], abductive and argumentation-based proof procedures [17] [9], etc, for ISE applications possibly involving inconsistencies.

AI has brought forward Default Logic [27], Circumscription [23], Truth Maintenance systems [24], Belief Revision [11], Diagnosis systems [7], temporal logics [35], etc. To say that CL, on which all of these formalisms recur, is a sufficient foundation, is blind against the fact that CL does not tolerate inconsistency at all. In fact, the cited approaches lack a common foundation and are thus imperiled to drift apart and finally into oblivion, unless there is a unifying foundation. We believe that the foundation to be laid for these approaches and applications needs to be paraconsistent.

In PhL, recent developments include Adaptive Logics (abbr. AL) [2], Inferential Erotetic Logic [36], Erotetic Search Scenarios [32] [37] and Socratic Proofs (SP) [33] [38] (cf. [12] for erotetic logics, EL, in general). In the sense of 4, Adaptive Logics is a full-fledged paraconsistent system. Among others, it reconciles hitherto antagonistic features of, on one hand, nonmonotonic reasoning, which is tolerant with regard to possibly conflicting conclusions, and, on the other, paraconsistent logic, which is tolerant with regard to conflicting premises.

With regard to ISE, we point out that AL, the mentioned EL systems and SP are inferential logic systems with computational interpretations, which exhibit remarkably like-minded similarities to abductive and argumentation-based procedures in the logic programming (abbr. LP) literature. (With regard to AL, this observation was made by Jeremy Carroll and this author during the 1st World Congress on Paraconsistency, 1997, and communicated in some email exchanges with others, but never analysed more closely, so far.) The mentioned LP proof procedures are characterized in 4.4 as paraconsistent. And a closer look at the cited PhL systems reveals a promising potential to describe, explain, predict, empower and operate reasoning processes in a paraconsistent manner with contradictions, conflicts and inconsistencies.

A tentative conclusion of this section could be that the time seems to be right for starting investigations into the objective of a unifying paraconsistent foundation of different approaches related to ISE.

4 Datalog as a paraconsistent foundation of ISE

In this section, we look into well-known *datalog*-based proof procedures which are widely used in databases, ISs and AI, assessing their potential as a point of departure for a paraconsistent foundation of ISE. To do so, we first define some criteria according to which the potential of paraconsistency can be objectively measured. To this end, we define in 4.1 and 4.3 some declarative and procedural notions of paraconsistency. In 4.2 we compare them to the historically first definitions of paraconsistency, and in 4.4 we discuss the definitions.

4.1 Definitions of Declarative and Procedural Paraconsistency

Usually, paraconsistency is defined declaratively, as non-explosiveness (i.e., ECQ does not apply). We are going to present a closely related procedural definition, and some refinements. By these definitions, *datalog*-based procedures of interest to ISE turn out to be paraconsistent.

Def. 1 (*declarative*): A deductive system S is paraconsistent if there is an inconsistent theory T and a well-formed formula (abbr. *wff*) F such that F is not derivable in T using axioms and inference rules in S.

Def. 2 (*procedural*): A proof procedure P is paraconsistent if there is an inconsistent theory T and a wff F such that there is no proof of F in T using P.

Def. 1 is standard. Def. 2 is not encountered in the literature, but largely corresponds to Def. 1, specializing the derivability relation in S of Def. 1 to running a particular proof procedure P. Clearly, Def. 2 entails Def. 1, in the sense that a proof procedure can always be thought of as a specialization and thus restriction of a deductive system, which may host a variety of different proof procedures.

Both definitions are privative, demanding a particular instance of incompleteness, but almost nothing "positive", nor much of a general property of S or P, respectively. Only the existence of a single pair (T, F) of a theory and a formula which do not behave classically (and might be practically irrelevant) is required. In the following subsection, the definitions are contrasted to more demanding requirements that had originally been postulated for paraconsistent systems at the inception of paraconsistent logic.

4.2 The Original Postulates of Paraconsistency

Jaśkowski originally postulated the following three properties for a logic calculus to be paraconsistent, in a declarative and informal manner.

(j1) Contradiction should not always lead to trivialization.

(j2) Practical inference should remain possible.

(j3) Axioms and inference rules should have intuitive justifications.

Postulate (j1) largely corresponds to Def. 1, when contradiction is equated with inconsistency and over-completeness with triviality, in the sense of the explosiveness of CL in the presence of inconsistency. Postulate (j2) seems to ask that restrictions imposed for satisfying (j1) should not hamper too many other deductions which are conveniently obtained in CL. Postulate (j3) seems to allude to properties such as simplicity and evidence, traditionally considered desirable for science in general.

Da Costa's hierarchy C_i $(1 \leq i \leq \omega)$ of paraconsistent logics can be seen as approaches to satisfy the conflicting postulates as expressed below (where, for $i < j$, C_i satisfies (dc2) better than C_j, while C_j satisfies (dc1) better than C_i).

(dc1) Tolerate inconsistency as much as possible.

(dc2) Conservatively approximate CL as much as possible.

Both sets of postulates ask more from a system to be paraconsistent than Def.s 1 and 2. So, the challenge is to find formalizations of (j1) - (j3) and (dc1), (dc2) which would yield paraconsistent systems with more desirable properties than the mere deficiencies asked for in Def.s 1 and 2.

A frequent postulate imposed on paraconsistent systems is that, in the consistent case, they behave as CL. That certainly is a conceptually sensible requirement. However, for computational purposes, there are no tractable inference procedures that have the full power of CL. Thus, the following postulate, asked for paraconsistent extensions Υ (say) of some CL-based IS S (such as, e.g., Horn clause logic), seems more reasonable.

(ex) Υ, when restricted to the consistent case, yields the same answers as S.

This postulate seems like a minimal requirement, but it makes sense when taking into account that most paraconsistent systems are obtained by restrictions imposed on CL which hamstring its power of inference. In fact, (ex) can be straightforwardly read as a concretion of postulates (j2) and (dc2).

4.3 Definitions, continued

The following definitions are meant as first steps in the direction of requiring more encompassing and more desirable formal properties of a paraconsistent proof procedure.

Def. 3: A proof procedure P is *fairly paraconsistent* if, for each theory T, there is a wff F such that there is no proof of F in T using P.

In all definitions, it is tacitly assumed that the language of T is expressive enough to include non-provable formulas and to allow the representation of inconsistency at all. Note that this is not the case for relational databases consisting of nothing but plain facts, but it is as soon as the database is required to satisfy consistency with regard to a set of integrity constraints represented in denial form.

Obviously, Def. 3 entails Def. 2. Note that Def. 3 does not explicitly refer to the consistency status of T. However, the defining property is required from each theory, hence in particular from inconsistent ones. That way, consistency is conceived merely as a special case of paraconsistency.

Although Def. 3 demands more than Def. 2, it still is content with the existence of a single non-provable formula, which however may very well be irrelevant in most cases. Possibly more relevant formulas are going to be considered in Def. 4, below.

The following definition focuses on resolution-based proof procedures. We assume that formulas be represented in clausal form, and the dependency between clauses be defined as usual, e.g., in [1]. For such a theory T and a formula F, let $dep(F, T)$ denote the set of clauses in T on which F depends, i.e., those clauses that are candidate input clauses in any attempt to prove F in T.

Def. 4: A resolution proof procedure P is *pretty paraconsistent* if, for each theory T and each well-formed formula F such that there is no proof of F in $dep(F, T)$ using P, there is no proof of F in T using P.

In Def. 3, only the existence of a non-provable formula F is required, while in Def. 4, the defining property is required for each non-provable formula F.

4.4 Discussion of Definitions

Under the assumption that the language of T is sufficiently rich, it is easy to see that Def. 4 entails Def. 3, which in turn entails Def. 2.

The general resolution proof procedure as defined by Robinson is not pretty paraconsistent, since it sanctions the inference of any formula from an inconsistent theory. However, the well-known SL resolution [15] (a specialization of the general procedure) is pretty paraconsistent: For a given goal to be refuted, SL considers as candidate input only those clauses on which the goal depends, and does not use disjunctive thinning, so that no irrelevant clauses are derived which would unnecessarily increase the set of clauses on which the goal depends. The proof that SL resolution is pretty paraconsistent is trivial, since the search space of any attempt to construct an SL refutation in T, rooted at the negation of F, is a subset of $dep(F, T)$.

Similarly, also the well-known *datalog*-based procedure of SLD resolution (when run on databases with integrity constraints in denial form) is pretty paraconsistent. Informally, the paraconsistent behaviour of SL and SLD resolution has been observed earlier [16]. Our definitions can thus be seen as an attempt to formalize those observations.

It is easy to see that also SLDNF resolution (SLD extended by the non-monotonic negation-as-failure inference rule) [21] and similar *datalog* procedures (which either compute a two- or three-valued completion semantics or a perfect or a stable or a well-founded model semantics) as well as abductive extensions thereof are pretty paraconsistent. Approaching the issue on the basis of definitions that are somewhat different from ours, similar conclusions about the paraconsistency of resolution-based CoL procedures with negation have been discussed in [25].

Moreover, we observe that the mentioned CoL procedures behave even better than what is demanded by Def. 4, e. g., when reasoning in inconsistent ISs with a high degree of interdependency of clauses. For instance, in a system with the single clause $p \leftarrow\sim p$ (which is inconsistent by the standard two-valued completion), no wrong answer is inferred by SLDNF for any query, even though no reasonable answer is obtained either for the query $\leftarrow p$, since SLDNF then loops. However, some abductive procedures (cf. [17]) terminate for this query, with the correct answer that p can not be proved; the same answer is derived for the query $\leftarrow\sim p$. Even the inconsistent integrity theory $\{\leftarrow p, \leftarrow\sim p\}$ (requiring that neither p nor $\sim p$ must be provable) will not lead to wrong answers in some abductive procedures (e.g., the one described in [5]), nor to any explosive behaviour. Thus, resolution-based procedures which are widely used in ISs, databases, AI and CoL are not only paraconsistent, but behave even better in the presence of inconsistency than what is required by Def.s 3 and 4, which are already more demanding than standard definitions of paraconsistency.

After all, SL, SLD and extensions thereof may be considered better paraconsistent inference systems than most established paraconsistent calculi, since the latter seem to be further removed from intuitionistic logic and CL than the *datalog* procedures: Resolution in general and its abductive extensions in particular make explicit use of LoC and the closely related *reduction ad absurdum* rule, while most other paraconsistent calculi have opted for abandoning the fundamental LoC and keeping the doubtful TND. For more technical details in support of this argument in favour of logic programming, the reader is referred to [6] and the cited references of related resolution-based proof procedures.

5 Conclusion

After presenting paraconsistency in its historical context, we have revisited and refined the usual declarative postulates for paraconsistency in procedural terms. We have seen that *datalog*-based procedures embody a practically viable kind of paraconsistency. In fact, it can be argued they do so more successfully than the systems proposed by the main protagonists of paraconsistency. Technical details have been spared for being consulted by the reader in the correspondingly cited references. In conclusion, we suggest that *datalog* and its extensions by integrity constraints and abduction should be considered as a point of departure for a thorough revision and replacement of established but deficient classical foundations of ISE by a new paraconsistent fundament.

References

1. S. Abiteboul, R. Hull, V. Vianu: *Foundations of Databases*. Addison-Wesley, 1995.
2. Adaptive Logics Home Page http://logica.ugent.be/adlog/al.html.
3. V. A. Bazhanov: Toward the Reconstruction of the Early History of Paraconsistent Logic: The Prerequisites of N. A. Vasiliev's Imaginary Logic. *Logique et Analyse* 41(161-163):17-20, 1998.
4. http://www-groups.dcs.st-and.ac.uk/∼history/Mathematicians/Bolzano.html.
5. H. Decker: An Extension of SLD by Abduction and Integrity Maintenance for View Updating in Deductive Databases. *Proc. JICSLP '96* 157-169, MIT Press, 1996.
6. H. Decker: Historical and Computational Aspects of Paraconsistency in View of the Logic Foundation of Databases. *Semantics in Databases* 63-81, Springer LNCS 2582, 2003.
7. R. Davis, B. Buchanan, E. Shortliffe: Retrospective on "Production Rules as a Representation for a Knowledge-Based Consultation Program". *Artif. Intell.* 59(1-2):181-189, 1993.
8. N. da Costa: On the Theory of Inconsistent Formal Systems. Notre Dame Journal of Formal Logic, 15(4):497-510, 1974.
9. P.M. Dung, P. Mancarella, F. Toni: Argumentation-Based Proof Procedures for Credulous and Sceptical Non-monotonic Reasoning. *Computational Logic: Logic Programming and Beyond* 289-310, Springer LNCS 2408, 2002.
10. http://www-groups.dcs.st-and.ac.uk/∼history/Mathematicians/Gentzen.html.
11. P. Gärdenfors, H. Rott: Belief Revision. *Handbook of Logic in Artificial Intelligence and Logic Programming* vol. 4, 35132, Clarendon Press, 1995.
12. D. Harrah: The Logic of Questions. In *Handbook of Philosophical Logic*, volume 8, 1-60, Kluwer, 2002.
13. J. van Heijenoort: *From Frege to Gödel*. Harvard University Press, 1967.
14. S. Jaśkowski: Propositional Calculus for Contradictory Deductive Systems. Studia Logica 24:143-157, 1969.
15. R. Kowalski, D. Kuehner: Linear Resolution with Selection Function. *Artif. Intell.* 2(3/4):227-260, 1971.
16. R. Kowalski: Logic for Problem Solving. North-Holland, 1979.
17. A. Kakas, R. Kowalski, F. Toni: The Role of Abduction in Logic Programming. *Handbook of Logic in Artificial Intelligence and Logic Programming* vol. 5, 235-324, Oxford University Press, 1995.
18. A. N. Kolmogorov: O principie tertium non datur. *Matematiceskij Sbornik (Recueil Mathématique)* 32, 1924/25.
19. W. Kneale, M. Kneale: *The Development of Logic*. Clarendon Press, 1962.
20. T. Kuhn: The Structure of Scientific Revolutions. University of Chicago Press, 1962, second edition 1970.
21. J. Lloyd: Foundations of Logic Programming (2nd edition). Springer, 1987.
22. [Lu] J. Łukasiewicz: *O zasadzie sprzecznosci u Arystotelesa: Studium Krytyczne* (On Aristotle's Principle of Contradiction: A Critical Study), 1910.
23. J. McCarthy: Circumscription - A Form of Non-Monotonic Reasoning. *Artif. Intell.* 13(1-2):27-39, 1980.
24. J. P. Martins: The Truth, The Whole Truth And Nothing But the Truth: An Indexed Bibliography to the Literature of TMS's. *AI Magazine* 11(5): 7-25, 1991.
25. P. Mascellani: Negation as Finite Failure is Paraconsistent. *Proc. IC-AI 2001*, 874-880, 2001.
26. http://plato.stanford.edu/entries/logic-paraconsistent/.

27. R. Reiter: A Logic for Default Reasoning. *Artif. Intell.* 13(1-2):81-132, 1980.
28. http://www.sis.pitt.edu/∼mbsclass/hall_of_fame/shannon.htm.
29. http://www.home.gil.com.au/∼bredshaw/stibitz.htm.
30. http://plato.stanford.edu/entries/tarski-truth/.
31. http://plato.stanford.edu/entries/turing/.
32. M. Urbanski: Synthetic Tableaux and Erotetic Search Scenarios: Extension and Extraction. *Logique et Analyse* 44(173-175), New Goff, 2001.
33. M. Urbanski: Computing Abduction with Socratic Proofs. Presented at VlaPoLo7, 2003. http://www.vub.ac.be/CLWF/VlaPoLo7/urbanski.pdf.
34. A. Urquhart: Many-Valued Logic. In Gabbay, Guenthner (eds): *Handbook of Philosophical Logic*. Kluwer, 71-116, 1986.
35. J. van Benthem: Temporal Logic. *Handbook of Logic in Artificial Intelligence and Logic Programming* vol. 4, 241-350, Clarendon Press, 1995.
36. A. Wiśniewski: *The Posing of Questions: Logical Foundations of Erotetic Inferences*. Kluwer, 1995.
37. A. Wiśniewski: Erotetic Search Scenarios. Synthese 134(3):389-427(39), Kluwer, 2003. See also *Erotetic Search Scenarios, Explanation and Abduction*, http://logica.rug.ac.be/censs2002/abstracts/Wisniewski.htm.
38. A. Wiśniewski: Socratic Proofs. *Journal of Philosophical Logic* 33(3):299-326, 2004. See also: A. Wiśniewski, G. Vanackere, D. Leszczyska: Socratic Proofs and Paraconsistency: A Case Study. http://logica.rug.ac.be/centrum/writings/pubs.php, Preprint 236.
39. http://homepages.inf.ed.ac.uk/wadler/papers/frege/.
40. http://www.hnf.de/museum/zuse_en.html.
41. W. Giloi: Konrad Zuse's Plankalkül: The First High-Level, "non von Neumann" Programming Language. IEEE Annals of the History of Computing 19(2): 17-24 (1997).

All URLs refer to web pages as viewed on April 17, 2005, at noon.