Introduction

The workshops in this series [1], [2] and [3] have been aimed at researchers and systems developers working on logic programming languages and their extensions, particularly but not exclusively those with a strong proof-theoretic flavour. Such extensions are important in meeting the needs of artificial intelligence for more expressive forms of knowledge representation, for better control of search and for better software engineering: but they also have strong links to research on logical frameworks and the approaches to system specification based on type theory. Newcomers to the subject are recommended to look at, for example, [4] as an introduction to one important approach.

The present volume covers a range of topics within this field:

Extensions of Horn logic: *Miller* proposes a means whereby modules may be used to structure large lambda-PROLOG programs. *Pinto* outlines a combination of functional and logic programming where both are seen as techniques for guiding an automatic search for a normal/cut-free proof. *Reddy* however shows that functional programming's higher-order features can be translated into a first-order logic programming setting without serious loss of expressive power; his argument is thus that higher-order extensions of PROLOG are in fact unnecessary, and that it is the concurrency and fairness issues that need to be more carefully addressed in order to allow proper translation of these higher-order features. *Boley* and *Dovier* address by different means the incorporation of finite sets into logic programming. *Ciampolini et al* consider problems of implementing a language with multi-head clauses on a transputer system

Knowledge representation: *Pearce* shows the relationship between disjunctive databases and Nelson's constructive logic with strong negation. *Alferes & Pereira* discuss two approaches — avoidance and removal — for dealing with the contradictions that may appear when negation is used in logic programs; the two approaches are shown to be equivalent. *Ribeiro & Porto* describe a language based on temporal propositional logic.

Linear logic: *Hodas* modifies his earlier work with Miller on the linear logic programming language LOLLI, allowing more sophisticated context management; *Zlatuška* shows how linear logic can model Concurrent PROLOG.

Partial inductive definitions: Several authors investigate properties or applications of Hallnäs' theory of partial inductive definitions (PIDs), a theory similar to the extensions of Horn logic allowing implications in goal formulae. *Eriksson* shows how to use a finitary version of the theory as a logical framework. *Falkman & Torgersson* discuss programming methodology in the PID-based language GCLA, concluding that different methodologies are appropriate for different sized programs and that GCLA needs a module system. *Kreuger* proposes a restriction on use of the axiom rule and explores the properties and applications of the ensuing calculi of PIDs. *Schroeder-Heister* shows that Eriksson's finitary rule (also proposed by Girard) for definitional reflection corresponds, in a suitable context, precisely to Clark's completion of a logic program.

Search spaces and search strategies: *Momigliano & Ornaghi* argue that one key feature of pure PROLOG (to be achieved in its extensions, and not in fact achieved in the negation-as-failure extension) is the regularity of the search space, in a sense similar to the meaning of confluence in term-rewriting theory. *Hinkelmann & Hintze* examine ways of estimating the costs of various proof strategies for PROLOG. *Abreu & Pereira* look at the possibility of intelligent pruning of the search space for programs in the Andorra Kernel Language. *Keronen* proposes a logic programming language where both knowledge about the problem domain and the search strategy can be expressed in logic. *Amrhein* shows that the models of a cumulative logic program form an equational variety, with applications to pruning of the search space in mind.

References

[1] Schroeder-Heister, P. (Editor): Extensions of Logic Programming, International Workshop, Tübingen, FRG, December 1989, Proceedings, Lecture Notes in Artificial Intelligence **475**, Springer-Verlag (1991).

[2] Eriksson, L.-H., L. Hallnäs & P. Schroeder-Heister (Editors): Extensions of Logic Programming, Second International Workshop, ELP '91, Stockholm, Sweden, January 1991, Proceedings, Lecture Notes in Artificial Intelligence **596**, Springer-Verlag (1992).

[3] Lamma, E., & P. Mello (Editors): Extensions of Logic Programming, Third International Workshop, ELP '92, Bologna, Italy, February 1992, Proceedings, Lecture Notes in Artificial Intelligence **660**, Springer-Verlag (1993).

[4] Miller, D.: Abstractions in logic programs, in: Logic and Computer Science (editor, P. Odifreddi), vol. **31** of APIC Studies in Data Processing, Academic Press (1990).