# Logic Programming and Lean Reasoning Systems (abstract)

Adam Meissner

Institute of Control and Information Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland.
Adam.Meissner@put.poznan.pl

**Keywords:** Colmerauer Session.

The term *lean reasoning system* denotes a relatively small program containing only basic mechanisms, essential for soundness and completeness of knowledge processing. Programs of this type obviously cannot solve hard inference problems. Nevertheless, this approach yields various benefits. In contrast to complex systems, lean reasoners are not hard to verify. Moreover, they can be easily modified and adapted to particular hardware platforms. They are also remarkably efficient in solving less difficult problems. It follows from a lower overhead for handling their internal parts than it happens for advanced systems. Furthermore, lean reasoning systems can act as convenient test-beds for comparison of various inference techniques, where the absolute efficiency is not as important as the relative one.

Lean reasoners are usually implemented in logic programming paradigm. Most often, they have a form of programs written in the Prolog language. It mainly follows from the fact that any execution environment of Prolog (e.g. its interpreter) can be regarded as a reasoning system for Horn clause logic. Hence, many elements of the Prolog language and its computation model (for example, operations on terms, SLD-resolution inference rule and the depth-first proof-search strategy, i.e. DFS) can be absorbed to the constructed reasoner, which significantly simplifies its structure.

In this paper we focus on one of best-known lean reasoners, namely on the system PTTP built by Stickel. We present our own results as well as some other works, which concern extensions, modifications and applications of PTTP. The system processes a knowledge expressed in FOL. It absorbs all basic inference mechanisms from Prolog and extends them to obtain a sound and complete reasoning procedure for full first-order logic. In particular, the unification of terms is enriched by the occur-check test, the SLD-resolution is turned into a model elimination rule and the depth-first proof-search strategy is augmented to depth-first iterative deepening search (DFID). In one of our early papers [3] we present a simple dialog system based on the system TALK created by Pereira and Shieber. The original system translates some subset of English language into FOL formulas. Then it narrows the result to Horn clauses and processes them by the standard Prolog interpreter. We replace this interpreter by the PTTP reasoner and show how it increases the question-answering power of the whole dialog system.

The system PTTP inspired also Lukacsy and Szeredi, the creators of DLog, which is a reasoner for the description logic (DL) with the language $\mathcal{SHIQ}$. It takes an advantage of highly-optimised Prolog database access procedures to perform ABox reasoning. The procedures are absorbed from the Prolog execution environment. In some aspects the DLog system can be seen as a simplified variant of PTTP. For instance, it uses a standard Prolog depth-first search instead of DFID search. This is correct since the calculus processed by the system DLog is decidable. However, we show in the paper [2] that adding the DFID strategy to DLog can notably improve the system performance for some types of queries. The strategy is added to the system by means of a simple metainterpreter, which makes use of certain mechanisms that are specific to SWI Prolog execution environment.

Finally, we consider an implementation of PTTP in so-called *relational programming* paradigm, which is a part of a programming model of the language Oz. The relational programming is closely related to the logic programming in Prolog. However, unlike in Prolog, the search strategy is not fixed in the execution environment but it is given as a *search engine* – a special object, which runs a program. Thus, the same program can be executed according to various search strategies. This is very convenient, especially in the prototyping and testing phase. A number of engines are available in standard Oz libraries. One of them runs a program in parallel on distributed machines. In the paper [1] we describe the PTTP system implemented as a relational program. We also present results of experiments aimed at estimating the work granularity and the speedup obtained by parallel processing.

## References

1. Meissner A., Introducing Parallel Computations to a PTTP-based First-Order Reasoning Process in the Oz Language, LNCS, Vol. 5138, 2008, pp. 359–364.
2. Meissner A., Brzykcy G., Reasoning with the Depth-First Iterative Deepening Strategy in the DLog System, LNCS, Vol. 7046, 2011, pp. 504–513.
3. Meissner A., Obrębski T., Prologowa technika dowodzenia twierdzeń jako metoda wnioskowania w prostym systemie dialogowym, w: Materiały III Krajowej Konferencji IWSE, Wrocław, 1997.