Curriculum Vitæ of
Alain Colmerauer

Last update, November 2007

Civil status

Alain Marie Albert Colmerauer, French, born January 24th, 1941, at Carcassonne, married with Colette Coursaget, three children, living at: 3 impasse des Iris, 13009 Marseille, France. Phone 33 4 91 40 65 47, e-mail alain.colmerauer@free.fr, web page http://alain.colmerauer.free.fr .

Diplomas

Computer Science degree from the engineering school, Institut Polytechnique de Grenoble in 1963. Doctorat d'état in Computer Science in 1967 at the University of Grenoble, under the supervision of Louis Bolliet and Jean Kuntzman, dissertation title: *Précédences, analyse syntaxique et langages de programmation*.

Positions

- 1963–1966. Researcher, in the team of Louis Bolliet, at the University of Grenoble .
- 1966–1967. Attaché de recherche CNRS, in the team of Louis Bolliet, at the'University of Grenoble .
- 1967–1970. Assistant Professor at the Computer Science department of the University of Montréal.
- 1970–1979. Professeur 2ème classe (Associate Professor) at the Faculty of Sciences of Luminy, University II of Aix-Marseille.
- 1979–1988. Professeur 1ère classe (Full Professor) at the Faculty of Sciences of Luminy, University II of Aix-Marseille.
- 1988–2000 Professeur classe exceptionnelle (University Professor) at the Faculty of Sciences of Luminy, University II of Aix-Marseille.
- 2000–2006 Professeur classe exceptionnelle (University Professor) at the Faculty of Sciences of Luminy, University II of Aix-Marseille, Institut Universitaire de France.

Supervision of research departments

- 1968–1970. Head of: projet de Traduction Automatique de l'Université de Montréal (TAUM).
- 1973–1985. Head of: Groupe d'Intelligence Artificielle (GIA), Centre National de la Recherche Scientifique and Université II of Aix-Marseille.
- 1991–1993. Head of: Groupe d'Intelligence Artificielle (GIA), Centre National de la Recherche Scientifique and University II of Aix-Marseille.
- 1993–1995. Head of: Laboratoire d'Informatique de Marseille (LIM), Centre National de la Recherche Scientifique and University de Provence and University de la Méditerranée.

Awards and honors

- Winner of la Pomme d'Or du Logiciel français 1982, an award from Apple France for a Prolog II implementation on Apple II, shared with Henry Kanoui and Michel Van Caneghem.
- Winner of the year 1984 of Conseil Régional of Provence, Alpes and Cte d'Azur.
- Award, Michel Monpetit 1985, given by the French Académie des Sciences.
- Chevalier de la Légion d'Honneur in 1986 (An honor bestowed by the French government recognizing outstanding contributions in various fields).
- Fellow of the American Association for Artificial Intelligence in 1991.
- Correspondant (Associate Editor) de l'Académie des Sciences in the area of Mathematics.

1963–1967 Syntactic analysis

I started research in 1963 at Grenoble while working on my doctoral dissertation. I was particularly interested in the parsing of context-free languages, in order to find the largest number of syntactic errors of a program in just one pass. I recall that the error-handling program was very general and even allowed one to shuffle a pack of punched cards, representing an Algol 60 program, and command the parser (using bottom up total precedence relations) to detect the displaced sequences [2].

1967–1970 Programming languages and automated translation

Since I was very much interested in syntactic analysis I turned my attention to a harder problem, namely the study of the syntax of natural languages. While working for the Automatic Translation Project of Montreal in 1968, I first wrote a general parser and generator for W-grammars [22], a formalism of A. van Wijngardeen for defining Algol68. It allowed the linguists of the project to write the different phases of a prototype of translation from English to French: English morphology, parsing of English, deep structure translation, generation of French, French morphology. After this experiment, in 1969 I developed the Q-systems, a formalism better suited for processing linguistic data [41, 11]. This formalism used rewrite rules of the type "the sub-sequence of trees of a given shape can be rewritten in the sequence of trees of another given shape". The rules were applied in parallel, using a non deterministic matching algorithm which took into account the associativity of the list concatenation operation. The Q-systems approach can be considered as the ancestor of Prolog [21]. The experience gained using the Q-systems reinforced my feeling that to suitably solve a problem it was necessary to develop high level programming languages, even if their execution time would appear impracticably slow at that time.

The Q-systems were used to construct an arsenal of programs (passes) in performing automatic English-French translations: the English morphology was written by Brian Harris, the English parser by Richard Kittredge, the deep structure translation by Gilles Stewart, the generation of French by Jules Dansereau, and the French morphology by Michel van Caneghem [42]. An industrial version of the Q-systems was also used a few years later to write the METEO system, which still produces daily translations of Canadian weather forecasts from English into French.

1970–1974 Birth of Prolog

Returning to France in 1970, I became increasingly more interested in making deductions from texts than in translating them into another language. With the help of Jean Trudel and Philippe Roussel I started studying automated theorem-proving, more precisely, the resolution principle of Alan Robinson. We came across a very interesting refinement of this method: SL-resolution, and, in June 1971, we invited his author, Robert Kowalski, who was researcher at Edinburgh to visit our research team in Marseilles. It was an unforgettable experience: Robert had a vast knowledge in automated theorem proving and his help was invaluable to us. Nevertheless, my aim at that time was not to create a new programming language but to describe to the computer in natural language (French) a small world of concepts and then ask the computer questions about that world and obtain answers. We wrote an embryo of such a system [43] and in that process the tool Prolog was developed. It was used for the analysis and the generation of French text, as well as for the deductive part needed to compute the answers to the questions [21].

The initial Prolog interpreter, written by Philippe Roussel in W-Algol, was improved one year later. Among various other features, I introduced the controversial but very useful search space cut operation. Philippe, after discussions with R. Boyer and J. Moore (at that time at Edinburgh) designed the first modern Prolog interpreter with shared data structures. It was programmed in Fortran by H. Meloni and G. Battani, postgraduate students at the time. That is the version of Prolog which was widely disseminated and subsequently improved: first in Edinburgh thanks to David Warren, then in Hungary, at the SRI of Stanford, and from there to Japan [21]

1974–1979 Work on natural language

I myself continued the work on natural languages, more precisely on semantics of the French language[3, 17]. I introduced several three-value logics to solve presupposition problems [16]. I became increasingly aware that the most interesting applications of my work were in natural language queries of data bases. I wrote a fairly large grammar of French which became the core of many applications. Veronica Dahl used it to write an interface with a Prolog program designed for computing configurations of Solar computers (made by the French Company Télémécanique). In her thesis she proposed a grammar of Spanish based on similar principles. Jean François Pique, Paul Sabatier and Chritian Giraud used the same grammar to write a question-answering system about French military hierarchy. David Warren and Fernando Pereira started from Veronica Dahl's work to develop question-answering systems in English and in Portuguese.

In 1976, under the strong advice of Michel van Caneghem we bought our first micro-computer: the Exorciser 6800 development tool of Motorola, with two floppy disks and 48k bytes of memory. The great challenge was to implement Prolog on this small machine. It was taken up by Henry Kanoui, Michel Van Caneghem and myself, by simulating a virtual machine using a floppy disk as slow memory.

1979–1982 Prolog II and the first micro-computers

The work done on the Exorciser encouraged us to further develop Prolog. The Apple II computers, purchased in 1977, now had 64k bytes of memory. Quite naturally, the idea came to our minds to implement an improved Prolog system using an inexpensive available computer.

I worked on the specification of this improved version of Prolog. I replaced the notion of unification by the notion of solving equations in a given domain. Using this new approach I introduced the concept of infinite trees and also of $\neq$ constraints: the new unification algorithm then became safe, without the costly *occur check*. It also became possible to check if two objects are unequal without using the search space cut operation [5, 23].

All this work gave birth to Prolog II [7] which would run effectively on an Apple II computer with a virtual memory on a floppy disk addressed by words of three bytes. We received an Apple France award for having developed this software.

1982–1990 Prolog III and the constraints

From October 1982 until October 1983 I spent a year in Paris at the Centre Mondial d'Informatique. I was relieved from the administrative duties of the University and I had the opportunity to start afresh new research on the extensions of basic principles of Prolog. In order to introduce infinite trees and the $\neq$ relation in Prolog II, I had been led to replace the concept of unification by the concept of constraint solving in a given domain with precise operations and relations. The obvious question became: Why not explore richer domains?

That is how Prolog III [10] came to existence. This language integrates at the constraint solving level: (1) a refined manipulation of trees, including infinite trees and a specific handling of lists, (2) a complete handling of two-valued Bolean algebra, (3) the processing of numerical values using infinite precision, including addition, multiplication by a constant and the relations $<, \leq, \geq, >$, (4) the generalized processing of the relations $\neq$ and of course $=$.

The development of Prolog III was not an easy task. (1) I first defined the major features of the language and the necessary algorithms for processing it. At the same time I conceived several examples of programs. (2) Michel Henrion, Frédéric Benhamou, Jean Marc Boï and Touraïvane, at that time PhD students [92, 93, 96], refined these algorithms and designed the modules for solving constraints in the different sub-structures involved in this Prolog. (3) Touraïvane integrated all these modules into a single module. (4) I finished the specifications of the language, adding among others the notion of delayed multiplication [20] and concatenation which Touraïvane had introduced.

Tasks (2) et (3) were performed in two years and at the end of 1987 we tested the first prototype of an interpreter Prolog III written in Pascal. It would take two additional years for our company PrologIA (established in January 1984) to commercialize a version of this prototype.

It should be mentioned that the development of Prolog III was possible through the strong financial support from the EEC and the Centre National d'Etude des Télécommunications.

1990–1996 Prolog IV and the constraints

It became clear to me that *constraint programming* consists of expressing a problem in terms of unknowns submitted to a constraint. Such a constraint has the form of a conjunction of elementary constraints and, more generally, the form of a first order formula involving operations and relations defined in a given domain. The function of the computer it to solve the constraint, that is to find the values which must be assigned to the free variables of the formula in order to obtain the value *true*. This is a unified vision of Logic Programming and the Mathematical Programming concept known in Operational Research.

According to this paradigm, I worked concurrently on constraint solving algorithms and on the development of complete programming systems using these algorithms. The result of all this work was Prolog IV [51], which became operational in July 96.

This language is characterized by a very large set of constraints: more than hundred predefined elementary constraints involving: (1) constraints on lists and trees, (2) numerical constraints handled in infinite precision by Gauss and Simplex type algorithms, (3) numerical constraints handled by narrowing and propagating floating point intervals, with the integers considered to be a special case of the real numbers and the Boolean values a special case of the integers.

The following example of constraint solving provides a good overview of the possibilities of the language:

$$\exists u \exists v \exists w \exists x \begin{bmatrix} \quad y \leq 5 \\ \wedge \ v_1 = \cos v_4 \\ \wedge \ \text{size}(u) = 3 \\ \wedge \ \text{size}(v) = 10 \\ \wedge \ u \bullet v = v \bullet w \\ \wedge \ y \geq 2 + (3 \times x) \\ \wedge \ x = (74 > \lfloor 100 \times v_1 \rfloor) \end{bmatrix}$$

becomes

$$y = 5.$$

In the above, x, y are real numbers, u, v, w are vectors, v_1, v_4 are the first and the fourth component of v and $(74 > \lfloor 100 \times v_1 \rfloor)$ is 1 or 0 according to the fact that 74 is or is not greater to the integer floor value of $100 \times v_1$.

Several years were necessary to write down the precise specifications of the language given in *Les bases de Prolog IV*, chapter 2 of the Prolog IV manual [51]. The two main problems were (1) to describe the very heterogeneous mathematical structure π_4 on which the language is built and (2) to clarify the different incompleteness of the constraint solving algorithms.

The chosen structure π_4 is the tree structure with 124 additional relations. The only operations are therefore the tree constructors. A large number of relations is used for expressing numerical operations, by representing a number by a tree of one node labelled by that number.

In order to characterize the incompleteness of the solving algorithms, I have isolated the 25 properties of the π_4 structure which are systematically used. These properties have been expressed in the form of first order axiom schemes and thus define the theory $\mathcal{T}_4$ in which the Prolog IV constraints are solved.

For the design of Prolog III and Prolog IV, I had to study and combine several algorithms including: Gaussian elimination, simplex, narrowing and propagation of intervals, solving equations and disequation on trees etc. I feel that the main problem now is the need to study deeply and separately well chosen constraint solving algorithms.

1996–2000 Constraints solving by intervals narrowing

We are essentially interested in solving conjunction of atomic constraints, by solving iteratively overlapping elementary sub-constraints and this until we obtain a fixed point. Every elementary sub-constraint is made from a central atomic constraint and a set of atomic constraints which give the range for every variable of the central constraint. The solving of an elementary constraint, often qualified *local*, consists in narrowing these ranges, which are generally intervals.

Combinatories problems examined I finished the supervision of the thesis of Ian Gambini [105] which was about the spliting of a square in small squares, of different sizes.

With Bruno Giletta we have worked on the problem in filling a box of dimensions $6 \times 5 \times 4$ elementary cubes by 12 pentaminos each one composed of 5 elementary cubes. It is there that the resolution of constraints by narrowing intervals makes sense. This work ended with the presentation of a thesis at the end of the year 2000 [107].

Sortedness constraint I am interested to solve *global* constraints of the form

$$\begin{bmatrix} (x_1, \ldots, x_n) \in r_n \\ \wedge \ x_1 \in A_1 \\ \ldots \\ \wedge \ x_n \in A_n \end{bmatrix}, \tag{1}$$

where the A_is are intervals of a totally ordered domain, like the set of real numbers or integers, and where r_n is a simple relation, but defined for unbounded values of n. Examples of r_n relations could be the set dif_n of n-tuples of distinct integers or the set $sort_{2m}$ of $2m$-tuples of the form $(x_1, \ldots, x_{2m})$ where the m-tuple $(x_{m+1}, \ldots, x_{2m})$ is obtained by sorting the m-tuple $(x_1, \ldots, x_m)$ in non decreasing order, etc.

Solving such a constraint (1) consists, not only of deciding whether it admits at least one solution, but also in computing, in the sense of inclusion, the smallest intervals X_i which can be substituted for the intervals A_i without changing the set of solution of (1). This is the same as computing the smallest Cartesian product of intervals $X_1 \times \cdots \times X_n$ which contains the set $r_n \cap A_1 \times \cdots \times A_n$ of n-tuples.

The objective is to find good global constraints which can be solved in polynomial time. With my mathematician colleague Noëlle Bleuzen-Guernalec we have developed an algorithm for solving the *sort* constraint in $\mathcal{O}(n \log n)$ time [12]. For example with $n = 2 \times 5$, this algorithm narrows the sequence of intervals $(A_1, \ldots, A_n)$ of constraint (1) as follows:

$$\begin{pmatrix} [0,13],[6,10],[10,11],[4,16],[4,6], \\ [1,3],[5,10],[6,9],[11,17],[10,15] \end{pmatrix}$$
$$\Downarrow$$
$$\begin{pmatrix} [1,3],[6,9],[11,11],[11,15],[5,6], \\ [1,3],[5,6],[6,9],[11,11],[11,15] \end{pmatrix}$$

I succeeded to solve this constraint, in reasonable time, with $2n$ equals up to 10 000 000.

From this algorithm we have inferred a way to solve also the dif_n constraint in $\mathcal{O}(n \log n)$ [53].

Approximation spaces While I was writing my PhD course, I felt the necessity to understand the concept of constraints solving by intervals narrowing. Thus I have introduced “approximation spaces” and “good” n-ary relation in order to surround its topological aspects. I have shown that r is a good n-ary relation if and only if each binary relation one can extract from r (by fixing $n-2$ variables to constant) is a good binary relation. I have presented my first results in three invited lectures [36]. One was the International Conference on Logic Programming, on the island of Cyprus, where I was surprised to get a special edition of the revue *Theory and Practice of Logic programming*, in honor to my 60th birthday [13].

2000–2004 Complete first-order theories

Full first order constraints on trees In many programming languages the notion of composite data is defined for representing various complex objects, like sequences, sequences of sequences, vectors, matrices etc. Essentially a composed datum is, either a simple datum like a number or an identifier, or a finite sequence of pointers to other composite data. Thus a composed datum can be considered as being a tree whose end nodes are labelled by simple data and whose intermediate nodes are labelled by identifiers.

It is thus important to be able to solve general constraints in the theory of trees. By *general constraints*, we understand non restricted first order formulae, that is formulae which are constructed with the quantifiers $\exists$, $\forall$, the logical constants *true*, *false* the connectors $\neg$, $\wedge$, $\vee$, the relation $=$ and terms made from variables and constructors.

Starting from the solving algorithms of Prolog II, Bich-Han Dao-Thi, one of my PhD students, has developed an algorithm which simplifies a first order formula p in a formula which is equivalent in the theory of infinite trees. This simplified formula is either the logical constant true or the constant false or a formula, true for some values of the free variables and false for some other values.

Thi Bich Hanh has conceived an algorithm expressed in 11 rewrite rules which ran on examples of constraints with 100 alternat quantifications $\exists, \forall, \exists, \forall, \ldots$ This work ended with a PhD end of 2000 [106] and an article [14].

We plan to study the complexity of this algorithm according to the different forms of the formula p. It is known that in the worst case this complexity is expressed by an iterated embedding of exponential functions, with a depth depending linearly on the size of p.

Full first-order constraints mixing trees and additive ordered rational numbers Later, with Khalil Djelloul in the frame of a thesis, we have combined the trees and the ordered additive rational numbers, in a complete theory [108].

Complete first-order theories I have started to establish a catalogue from the complete first-order theories which are the most interesting for computer science. I have presented this catalogue in two invited lectures, [34, 35]. I have also organized a workshop intitled ”Solving first-order constraints in various structures” in the “Centre International des Rencontres de Mathmatiques” at Marseille. Three themes were involved : first-order constraints resolution algorithms, axiomatisation in the first-order of interesting structurs, expressiveness power and complexity of the resolution of first-order constraints. Volker Weispfenning, University Passau, Andrei Voronkov, The University of Manchester Hoon Hong, North Carolina State University, Jean-Pierre Jouannaud, University Paris-sud, were among the participants.

2004–2006 Complexity of universal programs

Every year I ask my second year students to program a universal Turing. For this purpose I give them special software. In fact they do not program a complete machine – it would be too long – they rewrite one or two modules of a complete machine which I have beforehand tested and documented. My first machine was very slow and it was not conceivable to let it run on itself. As years passed, I conceived faster and faster machines and now they can run on themselves.

Starting from this work I defined and studied the complexity of universal programs. The obtained results were presented in two invited conférences and also in the ”Colloque annuel de l’Institut Universitaire de France” in March 2003 [38, 39, 28].

Developped softwares

- Parser and synthetiser of languages defined by W-grammars;
- The Q-systems;
- Prolog I (with Ph. Roussel);

- Prolog II (with M. Van Caneghem and H. Kanoui);
- Prolog III (with several PhD students and particularly Touraivane);
- Prolog IV (with PrologIA and particularly Touraivane).

Other activities

Being the first computer science professor in Marseille, I have developed this science "ex nihilo":

- by constantly taking care to have access to a computer;
- by forming younger colleagues;
- by supervising numerous theses;
- by creating in 1973 the "Groupe d'Intelligence Artificielle" (GIA), URA CNRS 816, at the Sciences Faculty of Luminy;
- by taking an active part in the creation of the "Laboratoire d'Informatique de Marseille" in 1994. It is an association of the CNRS and two universities: Aix-Marseille I University and Aix-Marseille II University. This laboratory involves more than hundred researchers (with the PhD students) and gathers most of the computer scientists from Marseille;
- by creating, as soon as 1975, a Computer Science "DEA" (PhD curriculum);
- by participating to the creation of number of courses in computer science (licence, maitrise, 2 DESS, département d'Informatique de l'école d'ingénieurs ESIL de Luminy) ;
- by having numerous contracts with the industry and the research institution; among others, I have obtained three ESPRIT contracts with two of them as principal contractor;
- by creating in 1984 the company PrologIA (with five of my colleagues). This company developed and sells Prolog II, Prolog III, Prolog IV and is involved in bank loans and air planes scheduling.

Papers in journals

[1] Alain Colmerauer. Notions d'opérateurs dans une grammaire "context-free", *RIRO*, no 2, 1967.

[2] Alain Colmerauer. Total Precedence Relations, *Journal of the ACM*, January 1970.

[3] Alain Colmerauer. Un sous-ensemble intéressant du français, in *RAIRO Informatique Théorique* 13, no 14, 1979.

[4] Alain Colmerauer. Sur les bases théoriques de Prolog, in *Groupe Programmation et Languages AFCET*, division théorie et technique de l'informatique, no 9, 1979.

[5] Alain Colmerauer, Henry Kanoui et Michel Van Caneghem. Prolog, Bases théoriques et développements actuels, in *TSI*, vol. 2, no 4 (AFCET-Bordas), August 1983.

[6] Alain Colmerauer et Jean-Franois Pique. About natural logic, dans *Logique et Analyse*, Nauwelaerts Printing S.A., 101-111, September 1985. Note : a second edition from [16].

[7] Alain Colmerauer. Prolog in 10 Figures, *Communications of the ACM*, vol. 28, num. 12, December 1985.

[8] Alain Colmerauer. Opening the Prolog III Universe, *Byte*, August 1987.

[9] Alain Colmerauer. Une Introduction à Prolog III, *Annales des Télécommunications*, 44, number 5-6, 1989.

[10] Alain Colmerauer. An Introduction to Prolog III, *Communications of the ACM*, 33(7): 68-90, 1990.

[11] Alain Colmerauer. Les systèmes-Q ou un formalisme pour analyser et synthètiser des phrases sur ordinateur. *Traitement Automatique des Langues*, Revue semestrielle de l'Association pour le Traitement Automatique des Langues, volume 33, Number 1–2, p 105–148, 1992. This is an old technical report from the University of Montréal which was never published before.

[12] Nolle Bleuzen-Guernalec et Alain Colmerauer. Optimal narrowing of a block of sortings in optimal time, dans *Constraints*, Kluwer Academic publishers. January, pp.85–118.

[13] Special issue in honor of Alain Colmerauer, on the occasion of his 60th birthday, *Theory and Practice of Logic programming*, volume1, part 6, November 2001.

[14] Alain Colmerauer et Thi-Bich-Hanh Dao. Expressiveness of Full First-Order constraints in the Algebra of Finite or Infinite Tress, dans *Constraints*, Kluwer Academic publishers. 2003, volume 8, issue 3.

Books and papers in books

[15] Alain Colmerauer. Metamorphosis grammars, dans *Natural Language Communication with Computers*, Lectures Notes in Computer Science n 63, edited by L. Bolc, Springer Verlag, 1978.

[16] Alain Colmerauer and Jean-François Pique. About natural logic, dans *Advances in Data Base Theory 1*, edited by H. Gallaire, H. Minker and J.M. Nicolas, Plenum Press, 1981.

[17] Alain Colmerauer. An interesting subset of natural language, in *Logic Programming*, edited by K. L. Clark and J. A. Tarnlund, Academic Press, 1982.

[18] Alain Colmerauer. Prolog and infinite trees, in *Logic Programming*, edited by K. L. Clark et J. A. Tarnlund, Academic Press, 1982.

[19] Frédéric Benhamou and Alain Colmerauer, editors. *Constraint Logic programming, Selected Research*. MIT Press, 1993.

[20] Alain Colmerauer. Naive solving of non-linear constraints, In Frédéric Benhamou and Alain Colmerauer, editors, *Logic Constraint Programming, Selected Research*. MIT Press, Cambridge, USA, 1993.

[21] Alain Colmerauer and Philippe Roussel. The birth of Prolog, dans *History of Programming Languages*, édited by Thomas J. Bergin and Richard G. Gibson, ACM Press/Addison-Wesley, 1996.

Conferences with proceedings

[22] Alain Colmerauer and Guy de Chastellier. W-Grammars, *Proceedings of the 24th National Conference* (San Francisco August 1969), New York: ACM Publication P-69, 1969.

[23] Alain Colmerauer. Equations and Inequations on Finite and Infinite Trees, paper presented as invited lecture, in *Proceedings of the International Conference on Fifth Generation Computer Systems*, Tokyo, November 1984.

[24] Alain Colmerauer. An Introduction to Prolog III, *Proceedings of the annual Esprit conference*, Bruxelles, September 1987.

[25] Alain Colmerauer and Philippe Roussel. The birth of Prolog. In *History of Programming Languages Conference, ACM SIGPLAN*, Cambridge USA, 20-23 April 1993.

[26] Noëlle Bleuzen Guernalec and Alain Colmerauer. Narrowing a $2n$-block of sortings in $\mathcal{O}n \log n$. In Proceedings of *Principles and Practice of Constraint Programming, CP97, Linz, octobre 97.*, edited by Gert Smolka, LNCS 1330, Springer Verlag, p 2–16, 1997.

[27] Alain Colmerauer and Bruno Giletta. Solving the three-dimensional pentamino puzzle, 2000.

[28] Alain Colmerauer. On the complexity of universal programs. In itshape Machine, Computations and Universality (Saint-Petersburg 2004). Lecture Notes in Computer Science, pp 18–35, 2005.

Invited lectures

[29] Alain Colmerauer. Naive concatenation. Invited talk, *Workshop on Principles and Practice of Constraint Programming*, Newport, Rhode Island, USA, April 1993.

[30] Alain Colmerauer. A legal framework for discussing approximate solving of constraints. In *Proceedings of INTERVAL 94*, St. Petersburg, Russia, March 1994. (Oral presentation with abstract in informal proceedings).

[31] Alain Colmerauer. Résolution approchée de contraintes par produits cartésiens de sous-ensembles privilégiés. In *Ecole de Printemps d'Informatique Théorique, Châtillon sur Seine*, May 1994. Invited lecture, no proceedings.

[32] Alain Colmerauer. Toward an ideal unified constraint domain, Invited lecture, no written text, (Pact 96, April 96, London) *Practical Application of Constraint Technology*, Proceedings published by The Practical Application Company Ltd, ISBN 0 9525554 25.

[33] Alain Colmerauer. Mixing constraints domain, the Prolog IV case, Invited lecture, no written text, *Frontiers of Combining Systems, First International Workshop*, Munich, March 1996, Proceedings edited by Franz Baader and Klaus U. Schultz, Applied Logic Series 3, Kluwer Academic publishers.

[34] Colmerauer Alain, Ngations et quantifications dans les contraintes, *Journes Francophones de Programmation en Logique et par contraintes (JFPLC 2000),* Marseille, June 2000.

[35] Colmerauer Alain, *Full first order constraints in various structure,* Confrence invite, Institute of Software, Chinese Academy of Sciences, Pkin, September 2000.

[36] Colmerauer Alain, Solving the multiplication constraint in several approximation spaces, *Seventeenth International Conference on Logic Programming ICLP-2001*, Paphos, septembre 2001 (and also University of Passau, November 2002).

[37] Colmerauer Alain, Cinq annes de recherche dans l'quipe de Louis Bolliet, 50 ans d'Informatique Grenoble, Grenoble, November 2002.

[38] Colmerauer Alain, Complexit d'un programme universel, confrence donne au *Colloque annuel de l'Institut Universitaire de France sur le thme de la complexit,* Ecole normale suprieure de Lyon, mars 2003. Aussi confrence invite, *Sminaire du Laboratoire d'Informatique Fondamentale d'Orlans,* Universit d'Orlans, October 2003.

[39] Colmerauer Alain, Complexity of Universals Programs, Jornadas de Programacin y Lenguajes, Alicante November 2003.

Internal reports

[40] Alain Colmerauer. Prcdences, analyse syntaxique et langages de programmation. Thse d'Etat. Universit de Grenoble, September 1967.

[41] Alain Colmerauer. Les systèmes-q ou un formalisme pour analyser et synthétiser des phrases sur ordinateur, internal publication no 43, Département d'Informatique de l'Université de Montréal, September 1970.

[42] Alain Colmerauer, Jule Dansereau, Brian Harris, Richard Kittredge, Gilles Steward and Michel Van Caneghem. TAUM 71, annual report of: projet de Traduction Automatique de l'Université de Monréal, January 1971.

[43] Alain Colmerauer, Henry Kanoui, Robert Pasero and Philippe Roussel. Un système de communication en français, preliminary report, Groupe de Recherche en

Intelligence Artificielle, University II Aix-Marseille, October 1972.

[44] Alain Colmerauer. Final Specifications for Prolog III. Esprit project P1219 (1106), Further Development of Prolog and its Validation by KBS in Technical Areas. Milestone II, February 1988.

[45] Alain Colmerauer. *Une introduction Prolog III*. LIM, 2000. Note : French version of [10].

[46] Alain Colmerauer. *Rsolution nave de contraintes non linaires*. Publication interne, LIM, 1992. Note : French version of [20].

[47] Alain Colmerauer. *Prolog IV Specifications*, Esprit project 5246, Reference Manual WP7/R25,TRD Prince, January 1995.

[48] Alain Colmerauer. Combining constraint domains. Esprit project 7195, Deliverable D2.11/3, Research Action Acclaim, June 1995.

[49] Alain Colmerauer and Philippe Roussel, *La naissance de Prolog*. Publication interne, LIM, 1996. Note : French version of [21].

[50] A. Colmerauer. *Les bases de Prolog IV.* LIM, 1996.

[51] F. Benhamou, P. Bouvier, A. Colmerauer, H. Garetta, B. Giletta, J.L. Massat, G.A. Narboni, S. N'Dong, R. Pasero, J.F. Pique, Touravane, M. Van Canegheme et E. Vétillard. *Le manuel de Prolog IV*. PrologIA, Marseille, June 1996.

[52] Noëlle Bleuzen-Guernalec and Alain Colmerauer. *Réduction d'un pavé de tris en $O(n \log n)$.* LIM, 1997. Note : French version of [26].

[53] Noëlle Bleuzen-Guernalec and Alain Colmerauer. *Narrowing a $2n$-block of sortings in $O(n \log n)$, application to the "all different" relation.* LIM, 1998.

[54] Noëlle Bleuzen-Guernalec and Alain Colmerauer. *Réduction optimale d'un pavé de tris en temps optimal.* Publication interne, LIM, 1999. Note : French version of [12].

[55] Alain Colmerauer and Thi-Bich-Hanh Dao. itshape Pouvoir d'expression des contraintes du 1ier ordre dans l'algbre des arbres finis ou infinis. LIF, 2003. Note : French version of [14].

[56] Colmerauer Alain, *Rsolution de contraintes par rduction d'intervalles,* Notes de cours : DEA d'Informatique et DESS de Mathmatiques et d'Informatique des Nouvelles Technologies, version 2004, Universit de la Mditerrane.

[57] Alain Colmerauer. *Sur la complexit de programmes universelles.* LIF, 2005. Note : French version of [28].

Supervised theses, University Aix-Marseille II

[58] Ph. Roussel. Définition et traitement de l'égalité formelle en démonstration automatique, thèse de 3ème cycle, May 1972.

[59] R. Pasero. Représentation du français en logique du premier ordre en vue de dialoguer avec un ordinateur, thèse de 3ème cycle, June 1973

[60] H. Kanoui. Application de la démonstration automatique aux manipulations algébriques et à l'intégration formelle sur ordinateur, thèse de 3ème cycle, October 1973.

[61] M. Bergman. Résolution par la démonstration automatique de quelques problèmes en intégration symbolique sur calculateur, thèse de 3ème cyle, October 1973.

[62] M. Joubert. Un système de résolution de problèmes à tendance naturelle, thèse de 3ème cycle, February 1974.

[63] J. Gispert. Etude de l'optimisation et réalisation d'un éditeur de textes paginés, thèse de 3ème cycle, June 1975.

[64] G. Battani. Mise en oeuvre des contraintes phonologiques, syntaxiques et sémantiques dans un système de compréhension automatique de la parole, thèse de 3ème cycle, June 1975.

[65] H. Meloni. Mise en oeuvre des contraintes phonologiques, syntaxiques et sémantiques dans un système de compréhension automatique de la parole, thèse de 3ème cycle, June 1975.

[66] J. Guizol. Synthèse du français à partir d'une représentation en logique du premier ordre, thèse de 3ème cycle, October 1975.

[67] V. Dahl. Un système déductif d'interrogation de banques de données en espagnol, thèse de 3ème cycle, November 1977.

[68] M. Rodriguez. Un système patient d'Aide à la conception JOB, thèse de 3ème cycle, October 1978. (Mainly supervised by Ph. Roussel.)

[69] Ph. Donz. Une méthode de transformation et d'optimisation de programmes Prolog. définition et implémentation, thèse de 3ème cycle, June 1979.

[70] P. Sabatier. Dialogues en Français avec un ordinateur, thèse de 3ème cycle, juin 1980. (Mainly supervised by R. Pasero.)

[71] Ch. Giraud. Logique et conception assistée par ordinateur, thèse de 3ème cycle, May 1980.

[72] L. Périchaud. Consultation en français d'une banque de données sur fichier et mise en place du système Prolog nécessaire, thèse de 3ème cycle, April 1981.

[73] J. F. Pique. Sur un modèle logique du langage naturel et son utilisation pour l'interrogation des banques de données, thèse de 3ème cycle, December 1981.

[74] P. Siegel. La saturation au secours de la Non-Monotonie, thèse de 3ème cycle, May 1981.

[75] G. Bossu. La saturation au secours de la Non-Monotonie, thèse de 3ème cycle, May 1981.

[76] H. Méloni. Etude et réalisation d'un système de reconnaissance automatique de la parole continue, thèse d'Etat, February 1982.

[77] J. P. Parcy. Un système expert en diagnostiç sur réacteurs à neutrons rapides, thèse de 3ème cycle, September 1982.

[78] M. Hileyan. Modélisation des fins de parties d'échecs, thèse de 3ème cycle, September 1982.

[79] M. Van Caneghem. L'Anatomie de Prolog II, thèse d'Etat, October 1984.

[80] H. Garetta. Un compilateur Modula II écrit en Prolog, thèse de doctorat, June 1985.

[81] S. Himbault. Un système expert en diagnostic de panne pour un réacteur nucléaire à neutrons rapides, thèse de doctorat, October 1985.

[82] C. Sedogbo. De la grammaire en chane du français à un système de questions-réponses, thèse d'Etat, June 1987.

[83] P. Siegel. Représentation et utilisation de la connaissance en calcul propositionnel, thèse d'état, July 1987.

[84] R. Picca. Implantation de Concurrent Prolog, thèse de doctorat, December 1987.

[85] H. Bellone. Concurrent Prolog : étude et utilisation, thèse de doctorat, December 1987.

[86] A. Vergara-Bracquemond. Exther, un système de diagnostic en échanges thermiques convectifs, thèse de doctorat, January 1988.

[87] C. Sabatier. Acquisition et interrogation de connaissances en langue naturelle, thèse de doctorat, April 1988.

[88] S. Grandcolas, Résolution d'équations sur les arbres et les listes, thèse de doctorat, May 1989.

[89] L. Hénocque, Un système logique pour le traitement du discours, thèse de doctorat, May 1989.

[90] J.L. Imbert, Simplification des systèmes de contraintes numériques linéaires, thèse de doctorat, May 1989.

[91] M. Rolbert, Résolution de formes pronominales dans l'interface d'interrogation d'une base de données, thèse de doctorat, May 1989.

[92] F. Benhamou, Le traitement des contraintes booléennes dans Prolog III, thèse de doctorat, November 1988.

[93] J.M. Boï, Le traitement des contraintes booléennes dans Prolog III, thèse de doctorat, November 1988.

[94] S. Coupet, Deux arguments pour les arbres infinis en Prolog, thèse de doctorat, November 1988.

[95] L. Oxusoff, Evaluation sémantique en calcul propositionnel, thèse de doctorat, January 1989.

[96] Touraïvane, La récupération de mémoire dans les machines non déterministes, thèse de doctorat, November 1988.

[97] J.J. Zotian. Prolog en informatique de gestion, thèse de doctorat, May 1988.

[98] Alfonso San Miguel Aguire. How to use symmetries in boolean constraint solving, thèse de doctorat, June 1992.

[99] Jean Luc Massat. Algorithmes énumératifs et résolutions de contraintes booléennes dans un langage de programmation logique avec contraintes, thèse de doctorat, March 1993.

[100] Jean-Louis Imbert. Les contraintes linéaires sur les nombres réels dans la cadre de la programmation en logique avec contraintes, habilitation, February 1993.

[101] Olivier Bartheye. Calcul de plans d'actions : des méthodes déductives vers les méthodes algébriques, thèse de doctorat, 1994.

[102] Jaam Jihad. Une étude sur les nombres de Ramsey classiques et multiples, binaires et ternaires, thèse de doctorat, February 1994.

[103] Jianyang Zhou. Calcul de plus petits produits cartésien d'intervalles, application au problème d'ordonnoncement d'atelier, thèse de doctorat, March 1997.

[104] Christophe Aillaud. Résolution de contraintes par analyse de parties convexes dans **R**, thèse de doctorat, July 1997.

[105] Ian Gambini. Quant aux carrs carrels, thse de doctorat, 1999.

[106] Thi Bich Hanh Dao. Rsolution de contraintes du premier ordre dans la thorie des rabres finbis ou infinis, thse de doctorat, 2000.

[107] Bruno Giletta. Placements des pentaminos par rsolution de contraintes, thse de doctorat, 2000.

[108] Khalil Djelloul, Thories compltes autour des arbres, (Complete theories around trees), thse de doctorat, June 2006.