ARTIFICIAL INTELLIGENCE IN WESTERN EUROPE

J. Pitrat and E. Sandewall (session co-chairmen)
W. Bibel, G. Huet, H.-H. Nagel and M. Somalvico

INTRODUCTION

Artificial intelligence research in Britain is generally well known to the American audience. Research in continental Western Europe is less well known, for several reasons:

it started later

barriers of language and habit decrease the exchange of information through newsletters and at conferences

for research on natural language understanding systems, which is a significant part of A.I., most researchers prefer to develop experimental systems in their own language. This makes it hard to publish the results in a foreign language

there are some differences in research approaches. In particular, resolution theorem-proving has a relatively stronger position in Europe than in the U.S.

The intention of this panel is to present an overview of the research that is being done in the field in Western Europe.

The present paper for the proceedings is a concatenation of surveys for each participating country. Hopefully the panel discussion at the conference will be used partly for presenting the surveys, and partly for comparing the experience of these countries.

The panel contains participants from four countries: France, Western Germany, Italy, and Sweden. It has not been possible to include all countries and all projects in each country, but we do believe that the survey is representative for the significant research projects in continental Western Europe, and that it covers most of them.

The contributors from the different countries have selected different focus and style for their surveys. Uniformity in these respects was considered to be neither desirable nor possible to achieve. Credit is given to the authors at the end of the paper.

FRANCE

Our account of A.I. research in France covers work at the following major centers:

- University of Marseille-Luminy (director: A. Colmerauer)
- University of Paris 8-Vincennes (director: P. Greussay)
- Institut de Recherche d'Inforraatique et d'Automatique (IRIA) at Versailles outside Paris (directors: G. Huet and G. Kahn)
- University Pierre et Marie Curie in Paris (directors: J. Pitrat and J.-C. Simon)

1. University of Marseille-Luminy

In 1971, A. Colmerauer formed an Artificial Intelligence group at the campus of Luminy of the University of Aix-Marseille. The main area of research is programming in first-order logic. A programming language based on first order logic, PROLOG, has been designed and implemented. It has been distributed to several A.I. centers, and the group in Luminy is maintaining and improving it.

A PROLOG instruction is basically a first-order Horn clause. Certain predicates are evaluable, and the search is guided by special marks between literals. The PROLOG interpreter is based on an efficient linear resolution theorem prover.

Various applications of the PROLOG system are under development in Luminy, in particular:

A natural-language understanding system has been implemented for a substantial subset of the French language. The meaning of sentences is expressed using typed lambda calculus terms, which are analysed by so-called metamorphosis grammars, coded as PROLOG programs.

A symbolic manipulation system is under development. It is capable of doing integration, trigonometry and computation of limits in interactive mode, on expressions with several unknowns. The whole system is written in PROLOG.

2. University of Paris 6-Vincennes

The artificial intelligence group here has designed a version of LISP, called VLISP, which has several attractive features. It is based on very powerful control structures, most notably the concept of filters, which could be characterized as pattern-matching-controlled continuations. VLISP has been implemented on several computers. The implementation effort has been focused on efficient implementations on mini and microcomputers.

A new formalism for modeling various deductive processes, called CANON, is under development. It is based on the concept of meta-pattern-matching.

Among the applications developed in Vincennes, let us quote a programmer's assistant system for understanding LISP programs written by a novice programmer. The system is able to detect and correct various syntactic bugs, and a few semantic ones.

IRIA

In 1976, G. Huet completed his these d'Etat on unification in languages of order 1, ?, ..., w, which provides a general framework for the study of pattern-matching problems. This work has applications in automating program transformations expressed as schema rewriting rules in a higher-order language. More recently, he worked on theoretical properties of simplification systems which are useful for the efficient mechanization of equational theories.

Concurrently, G. Huet and G. Kahn direct a project concerned with the conception and implementation of a programmer's assistant. This system, called MENTOR, is language independent, and based on a general notion of abstract syntax. The programmer does all phases of programming under the supervision of the system, and communicates with MENTOR using a specialized programming language operating on the abstract syntax. Powerful pattern matching operations are provided. The system is currently operating on PASCAL programs, and is itself entirely written in PASCAL. It is used to maintain and improve itself. A high-level optimization package for PASCAL is under development, and a symbolic interpreter is under completion. Further developments will focus on enriching MENTOR's programming language with coroutine-like control structures.

4. University Pierre et Marie Curie, Paris VI

The two main groups working on Artificial Intelligence are directed by professors Simon and Pitrat. Simon is also working in pattern recognition, which is not the subject of this paper. There are also some other researchers working independently. T shall describe the work done in four main areas: learning, theorem proving, applications to computer science, and natural language understanding, without specifying to which group each researcher belongs.

A• Learning and inference

J. Pitrat has realized a program [1] which learns to play chess. One of the main problems in chess is to choose what moves are to be considered. The program learns to generate a set of plans for any position from games which are given to it as data. It tries to understand why a move has been successful. If it understands, it generalizes the characteristics which are necessary for the success. The program has learned effectively to find plans, but as it had difficulties to use these plans, a program was written [2] which does not learn, but uses plans efficiently.

C. Lemaitre [3] wrote a program simulating a robot learning to structure its world. In a complex world, a robot with a small memory learned to recognize remarkable places and to connect them. So, when it decided to go somewhere, it could move there quickly. A man learning to move in an unknown town has a similar problem to solve.

C. Roche's program *[4]* learned to create operators useful for solving pattern recognition problems. For instance, it received patterns which belonged to one of two sets: in one set, two points are close and in the other the two point are distant. Initially, it received the arithmetic operators: plus, times etc. It was able to find the Euclidean distance for separating the patterns in the two given sets.

0. Carriere *[5]* wrote a program which solves psychological tests used for measuring intelligence. Tests such as sequences of numbers, letters, dominoes to complete, matrices with an empty square, and others were solved. The first step is to find the relation between the elements of the known sequence for inferring the missing element.

M. Jouannaud [6] studied the automatic inference of LISP functions from a finite number of examples. The program can often infer a recursive function from one example. If it can't, it asks for new examples and builds it by steps. With Treuil and Guiho [7] he also wrote a program learning some arithmetic laws.

M. Kodratoff extended Sommer's results and wants to implement them in a program. Specifically, he increases the number of recurrence relations and proves the existence of the function when the primitive recursive relations are more complex.

B. Theorem proving

10 theses [8,9,10,11,12,13,14,15, 16,17] were made in this area and others are on the way. T will not present all these works, but only give the general idea which guide them. 1 do not believe that theorem proving is an efficient method for synthesizing programs, understanding natural language or solving robot problems. So, the goal of these theorem proving programs was only to prove mathematical theorems. Mathematics will probably be one of the first areas where programs will be as good as humans. But it is necessary to examine how the mathematician is working. In some areas, there are generally few choices for a human being in a given situation. Only one or two possibilities are a priori worthwhile. But there is a great variety of situations. M Vivet is studying how a program can use a set of methods for solving mathematical problems.

Another idea is to use a graph for helping the program. Some part of the fomula may be coded in the graph. For instance D. Pastre [16] replaces relations by arcs of a graph. The rules modify the formulas and the graph. Many steps become obvious when we consider the graph. The formulas are shorter.

Another use of a graph is to have a convenient storage. M. Buthion [17] wrote a program solving compass and straight edge construction problems in plane geometry. Unlike Gelernter's system, the diagram is not used for verifying whether a property is true. But the diagram stores certain facts,e.g. that some points are on the same line, or on the same circle.

We think that it is important to study the mathematician. Once, as we were interviewing, a mathematician said that he never used the usual notation for his theory, except when he was reading or publishing a paper. He himself worked with a representation which was not very clear, but entirely different from those used in the books. In such cases, a program now solves problems completely differently from how the mathematician does it. It is then normal that the program's performance is not so good, and an interesting problem is to study the representations really used by the mathematician, not when he is publishing, but when he is proving theorems.

C. Applications to computer science.

A first approach is the realization of programs which process programs which for various reasons have already been written. Pitrat wrote for instance a general game playing program [18]. The rules of a specific game were given as two separate programs written in a special language: one of them generated legal moves, and the other checked whether a given position was a winning one for either player. If the general program is to play well, it must be able to analyze the rules of a game for extracting useful information, particularly what conditions are necessary if a move is to be legal.

B. Falvigny [19] wrote a program for finding anomalies in a program; an example of an anomaly is the existence of an instruction such that, if it is executed, the program loops indefinitely. An anomaly is not always an error, but it indicates that, somewhere in the program, something is probably not correct.

A. Adam and J.P. Laurent [20] are writing a program to correct a student's program. This program finds which instructions are false and which instructions could be placed instead. For this, it compares the statement of the problem with the program given by the student.

Another application is program synthesis. Usually, we try to define the problem in a descriptive language. For instance the program of A. Adam [21] generates programs which check whether a graph has some property; the program of P. Gloess [22] generates programs translating a formula written in some notation to the equivalent formula written in another notation. In that case, the program receives the formal definition of both notations. F. Garijo's program [23] generates a program computing a function defined by a set of formulas. M. Masson's program *[2k]* generates programs checking whether a tree has a given property.

J.L, Lauriere's program [25] receives as data the descriptive statement of a combinatorial problem (with or without optimization). The data of a particular problem is given with the statement. The program first finds a solution, then, if optimization is needed, it finds the optimal solution and proves that this solution is optimal.

The method used by the prograu changes with the data. If the data includes oert&in values, the method used by the program works only because of these special values. For practical problems, its results were sometimes as good as those of programs which were written for this particular problem.

D. Understanding natural language

We want to understand how a program can understand. One method for finding this is to write a program understanding a Btory, then answering questions on this story. G. Sabah [26] and C. Loyo Varela [27] give the program a story in French. The program's goal is to find the motivation of the individuals who acts in this story. It uses pragmatic rules to find their motives.

Other programs answer questions in French on a blocks world (F. Rousselot [28]), in Spanish on the same domain (M.F. Verdejo [29]), in Arabic on the constitution of Lebanon (B. Wakim).

Another method for the same goal is to write a program solving problems given in a natural language: electricity exercises in French (M. Rousseau [30]) or mechanics exercises in Arabic(M. Gheith). The last method is to write a program which paraphrases. Pitrat is writing such a program for the French language: annotations of chess games.

We are mainly interested in the representation of the meaning of the sentences and to find what ir formation is necessary for understanding. Pragmatics have a great importance in our approach, but it is necessary to give the program a huge amount of information.

Working in another direction, but independently, D. Coulon and D. Kayser [31] have written a program learning to identify answers to a given question expressed in a natural language. From a sample of such questions, the program builds rules, and with these rules, it can classify the new answers. The rules are built using some elementary operators. This program is aseful for teaching machines.

GERMANY, FEDERAL REPUBLIC OF

For quite some years there has been research in Germany in areas which are usually considered part of artificial intelligence; but it was not until 1975 that the scientists working in these areas came together to identify themselves as a group with common scientific interests. Since then quite a bit has achieved.

- The group officially became a special interest group ("Fachgruppe") of the German association for informatics (Gesellschaft fur Informatik - GI) and, like other areas in informatics, it is represented within the GI by a committee ("Fachausschuss") whose members are scientists working in the field of A.I. and of pattern recognition. Those from A.I. are W. Bibel, J. LaubBch, H.-H. Nagel (chairman), and G. Veenker.

- A German A.I.-newsletter with similar intentions like the SIGART- or AISB-newsletter now appears regularly on a quarterly basis and is sent to about 200 addresses (currently edited by W. Bibel).

In this relatively short time of somewhat more than two years five meetings or workshops exclusively devoted to A.I.-research have been organized, not counting contributions to national conferences with A.I.-sections.

- An international conference on A.I. will take place in the third week of July 1978 in Hamburg organized by the GI in co-operation with the Society for the Study of Artificial Intelligence ana Simulation of Behaviour (AISB) from Britain. Local arrangements will be the responsibility of H.-H. Nagel from Hamburg. Dr. Derek Sleeman from the computer based learning project, University of Leeds will act as program chairman.

According to that relatively short history it is not surprising that there is no real center of A.I.-research nor any really big A.I.-project in Germany. The activities are scattered over many places, with some clustering taking place at the universities of Hamburg and Karlsruhe.

The scientific activities mainly concentrate on two areas of research: processing of naturally represented information (vritten natural language and vision) and automatic deduction. The common future perspectives for both are probably knowledgeable, human oriented programming systems. In the following we subdivide these areas and list names of people working in the respective subareas and their affiliation (which is normally "Institut fur Informatik, Universitat <name of city>", if not stated otherwise). This list is based on the necessarily incomplete information which is at the authors' disposal and therefore cannot be exhaustive.

1. Processing of naturally represented information
   a) natural language processing, syntax

semantics, translation
b) scene analysis
c) cognitive systems; formal representation of knowledge, semantic nets
d) (linguistic) question answering systems
e) (procedural) problem solving

There is a large image analysis group at the Forschungsinstitut fur Informationsverarbeitung und Mustererkennung (FIM) at Karlsruhe working with Kazmierczak. Their work has a large overlap with scene analysis (b, c): Bohner, Rocker, Scharf, Stiess.

A number of people working in this area are in Hamburg: Bertelsmeier, Kemen, Nagel, Neumann, Radig (b, c); Schefe, Wahlster, Wittig (a, c, d, e); Boley (c), and several others. At Stuttgart there are Laubsch (a, b, c), Hanataka (b), Krause (a, c). At TU Miinchen work J anas, Ripken, Schwind (a, c); at Saarbriicken Weber (a). Fischer (c) at the Bildungstechnologisches Zentrum, Darmstadt, Gorz (a, c) at Erlangen. Brecht and others (c, e) work at the Institut fur Deutsche Sprache, Mannheim and Bonn. Klaczko (c) at Frankfurt. Scene analysis (b) is pursued by Foith at the Institut fur Informationsverarbeitung in Technik und Biologie at Karlsruhe.

Automatic deduction

a) theorem proving in logical systems
b) program synthesis/analysis/verification
c) deductive question answering systems

The strongest group working in this area has its center in Karlsruhe: Deussen, Siekmann, Wrightson (a); Raulefs (b) and others; at Ronn work Darlington (a, b, c); Forning, Veenker (a); at Aachen Richter (a); at TU Miinchen work Bibel, Schreiber (a, b); v. Henke (b) works at the GMD, St. Augustin; Winterstein (a) at Ka:serslautern; at TU Berlin work Bergmann, Noll (a); Schneider, Konrad (e). Kudlek (a) works at Hamburg.

3. Chess, etc.

There are several game playing systems. In a tournament held in 1975 the most successful German chess playing system was one developed by Zagler and others at the TU Miinchen.

U. Miscellaneous

A surprisingly efficient system for solving differential equations has been developed by Schmidt, Bonn. - A formula manipulation system was designed by Steinbruggen, TU Miinchen. - AI methods are involved in a system for solving chemical problems which is being developed by Ugi and Brandt, Organ, chem. Labor, TU Miinchen.

In the area of automatic deduction a lot of work has been invested in the theoretical foundations (such as development and comparison of various complete theorem proving methods especially those based on natural deduction system, as well as semantics and representation of programs). But now, as software for string processing at the computing centers in Germany becomes more comfortable, there is a tendency towards building efficient, specialized, practicable systems rather than devloping further (complete) theoretical methods. But obviously most groups are still too small to provide the manpower necessary for the desr'gn and implementation of really powerful yystem. However, the perspectives for the near future seem to be not too bad in that respect.

Work in speech understanding is still pursued as part of pattern recognition. In contrast to this, people working in image analysis seem to be more aware of AI implications for their work. Although the larger image analysis projects at FIM/Karlsruhe and the Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt (DFVLR) Oberpfaffenhofen cannot yet be considered as AI projects, it would not be surprising to see major ventures into this area in the future.

ITALY

The survey of A.I. research in Italy has unfortunately been delayed in the mail.

SWFDN

Artificial Intelligence research in Sweden first developed in the Computer Science departments of some universities. In the last few years, some A.I. has been done in Linguistics and Prychology departments as well.

Uppsala and Linkoeping Universities. The Informatics Laboratory ("Datalogilaboratoriet") at Uppsala University was formed in 1970. In 1975 the major part of the group moved to Linkoeping University, while a minor part remained in Uppsala. The two groups have therefore similar profiles and will here be described together.

The group in Uppsala was formed by a few people who had worked on typical A.I. problems such as data-base structures for question-answering systems, heuristic search, and high-level languages. However, in specifying the area of research for the new group, we chose to invent the term (technology for) small data bases, rather than artificial intelligence.

In practice, "small data bases" has a considerable overlap with artificial intelligence. The data base of a Q.A. system is a typical small data base. In general, small data base technology is concerned with data bases where the conceptual complexity of the data and/or the operations that are to be performed on them are the major problems, in contradistinction to conventional data bases where the complexity of information is small, and the volume of data and economy of operation are the dominating problems. LISP-like languages, and very-high-level languages based on LISP, are software tools for small data bases. Natural language would probably for many purposes be an ideal means of communication with a small data base. The term small data bases is perhaps equivalent to "knowledge-based systems" that is now often used internationally.

The significant difference between "small data bases" and "artificial intelligence" is that the former term implies a stronger emphasis on software tools (such as programming languages) and on applications, and that it avoids the science-fiction-like overtones which have been both an attraction and a detriment for A.I. In general, we believe that our term has been useful. However, it could clearly not be used in a country where "big" is a positively loaded word, and "small" is negatively loaded.

The following are some of the major projects that have been concluded or nearly concluded in these groups:

- Implementation of an INTERLISP system for IBM 360 and 370 series computers (Jaak Urmi et al, Uppsala)
- Design of special hardware for a LISP system (Jaak Urmi, Linkoeping)
- Work on data bases for natural language understanding system, based on the idea chat predicate calculus or an extension thereof should be used as a specification language for both the data base and the storage and retrieval operations. Predicate calculus was not used for the representation of the NL informaoion, but rather as the notation wherein the representation of various semantic constructs was expressed. In a sense it was used as a programming language, although one which was not directly implemented, but instead one which served as the abstract program from which a concrete program in a concrete programming language (such as LISP) was written.
  (Erik Sandewall, Uppsala)

- Work on comprehension of natural language texts that describe processes, through implementation of a system which accepts descriptions of patience (solitaire) games in natural Swedish, analyzes the text, and performs the game. The system has been designed for real natural language texts, taken from books from a bookstore, rather than for constructed examples. The domain is interesting because the games are processes that involve conditional branches as well as loops. Since each game description is really an algorithm, the sublanguage used in describing games can be viewed as a natural programming language. The work therefore has a potential interst for other branches of computer science as well.
  (Mats Cedvall, Uppsala)

- Implementation of a compiler for ATN grammars. This work was done in the early stages of the patience game project, as a development of tools, achieved speed-up with a factor of ten in the IBM 370 INTERLISP.
  (Mats Cedvall, Uppsala)

The following projects are less directly related to artificial intelligence, but are natural consequences of the .smal3_ data__t)a:;e_ profile:

- Work on partial evaluation as a programdevelopmenu technique. This work has been done iteratively by designing a partial evaluator, using it for a number of applications, and revising the partial evaluator. The recent version of the system has been able to successfully partially-evaluate LISP code that had been written by an expert LISP programmer without previous knowledge that his program was going to be used in the experiment.
  (Anders Haraldsson and others, Uppsala and Linkoeping)

  Compiler for partial evaluator. In conjunction with the work on partial evaluation, there has been work on a partial-evaluator-compiler, which transforms a general-purpose program to a corresponding program generator, and on methods for proving that a proposed use of these programs is correct.
  (Erik Sandewall and others, Uppsala)

  A method for formal definition of the semantics of programming languages, demonstrated through a definition of Simula. The definition is an idealized compiler, and is significant because it is both effe£trve (it exists as a LISP program, and has been used to compile and run a systematically designed test batch of Simula

programs, which is significant for debugging the definition), and legible (the report describes how a number of real questions about Simula semantics are answered by referring to the formal definition).
(Mats Nordstrom, Uppsala)

An auxiliary system for design of man-machine dialogues. The system represents the dialogue as a network, where each node represents an interaction, and is associated with a typeout from the system, a syntax for possible user responses, indications for which node to go to next, etc. The system, called IDECS, contains an interpreter for the network with advanced facilities such as backtracking and undoing; an interactive editor on the network, and of course facilities for entry and presentation of the conversation network. The system was designed in response to a practical need, and enables one to let an end user try his hand at the proposed dialogues, find out what details have to be changed, make the change at the spot, and let the final user try again. This system was developed in a cooperation project with the Swedish Bureau of Statistics and has been used in several application projects.
(Sture Hagglund and others, Uppsala and Linkoeping)

All of the last three projects relied very heavily on the use of LISP as an implementation language, and probably could not have been completed if a conventional language had been used. They indicate that interactive programming technology in LISP is a significant spin-off result of A.I. research.

The projects that have been described represent a sample which characterizes the profile of the activities. A few additional projects should also have been described but have had to be omitted because of space restrictions.

Stockholm University. Sten-Ake Tarnlund and a few co-workers have worked since around 1970 with resolution theorem-proving. After initial work on implementation and experimental use of a theorem-prover on a mini-computer, the group has recently worked in the area of "using predicate calculus as a programming language". This work has included the implementation of several well-known algorithms in this language, an analysis of the properties of the language, and correctness proofs of algorithms expressed in the language.

Research Institute of National. Defense, Stockholm. A group headed by Jacob Palme undertook a project in the period 1968-73 to implement a fairly efficient question-answering system for a simplified and somewhat un-natural English. The project resulted in a general-purpose parser system and a "semantic net" type data base that was quite sophisticated considering its efficient implementation (in PL/360). There was also an experimental implementation with Esperanto as the input language.

The projects that have been described so far have been performed in a computer science environment. There are also a number of projects which were performed in other milieus. In 1975 a common interest group comprising people of these different backgrounds was formed.

Stockholm University, department of Linguistics. Carl-Wilhelm Welin has studied the problem of "understanding" natural Swedish through implementation of a system which accepts weather bulletins. These bulletins, published from the central weather bureau, are in Sweden customarily written in a sublanguage, which contains a large number of stereotypes and which at least initially appears to be sufficiently limited to be idea] as a test environment for a system endowed with non-trivial understanding.

Stockholm University, department of Psychology. Anders Eriksson has done work on problem solving methods in the context of the 8-game (reduced 15-game) Protocols of human subjects solving 8-puzzles were collected, and used as the basis for a model of problem-solving. The derived model uses production schemas and has a GPS-like flavor. The work was presented as a thesis. Additional A.I.-oriented projects have been started at the department .

Uppsala University, Computing Center. Anna-Lena Sagvall Ilein directs the development of tools for natural language processing, oriented mostly towards the needs of the language departments of the university and the computing center's speciality, medical data processing. A system for morphological analysis of Russian was the first project, and has been operational since 1973. A current project with a distinct A.I. flavor is an experimental system for understanding authentic medical reports on medial column fractures in the hip, written in Swedish. The system translates each sentence into a domain-specific internal representation. Work on detection of conflicts in the knowledge base comes next. - Other projects in the group are a chart-analysis parser according to Kay, and a system for inflectional analysis of Finnish.

REFERENCES - FRENCH CONTRIBUTION

[1] J. PITRAT: A program for learning to play-chess. Pattern recognition and artificial intelligence. Edited by Chen, Academic Press, 1976, p. 399-419.

[2] J. PITRAT: A chess program which uses plans. Artificial intelligence.

[3] C. LEMAITRE: Problemes de planification et apprentissage dans le cas d'un programme de simulation de robot. These de 3 cycle, Paris VI, *29-5-74.*

[k] C. ROCHE: Generation of recognition operators by stochastic and heuristic learning. Computer oriented learning processes. Edited by J.C. Simon. Nato Advanced Study Institute series. Noordhoff 1976, p. 237-255.

[5] 0. CARRIERE: Realisation d'un programme heuristique qui resout des tests psychologiques de mesure du facteur G. These de 3 Cycle, Paris VI, 19-6-73.

[6] J.P. JOUANNAUD, D. GUIHO, J.C. SIMON: Inference of functions with an interactive system. Machine Intelligence 9.

[7] J.P. TREUIL, G. GUIHO, J.P. JOUANNAUD: Une metnode d'apprentissage de concepts. Panorama de la nouveaute informatique en France. AFCET, 1974, Tome 1, p. 77-89.

[8] J. PITRAT: Une programme de demonstration de theoremes. Dunod, 1970.

[9] J.L. DELHAYE: Datal: Un program de demonstration automatique de theoremes. These de 3 Cycle, Paris VI, 9-4-70.

[10] J.P. LAURENT: A program that computes limits using heuristics to evaluate the indeterminate forms. Artificial intelligence, Vol. 4, N° 2, Summer 1973, p. 69-94.

[11] M. VIVET: Un programme qui verifie des identites en utilisant le raisonnement par recurrence. These de 3 Cycle, Paris VI, 29-22-73.

[12] R. DALLARD: Presentation d'un programme de demonstration de theoremes d'arithmetique. These de 3° Cycle, Paris VI, 24--4-74.

[13] M. GRANDBASTIEN: Un programme qui resout formellement des equations trigonometriques par des procedes heuristiques. These de 3 Cycle, Paris VI, 14-6-74.

[14] A DURAND: Un programme de demonstration d'exercices d'algebre. These de 3 Cycle, Paris VI, 13-6-75-

[15] A. HERTZ: Programmes de demonstration de theoremes formulables en logique des predicats du 1er ordre avec egalite. These de 3 Cycle, Paris VI, 2-12-75-

[16] D. PASTRE: Demonstration automatique de theoremes en theorie des ensembles. These de 3° Cycle, Paris VI, 22-10-76.

[17] M. BUTHION: Un programme qui resout formellement des problemes de construction geometrique. These de 3° Cycle, Paris VI, 13-6-75.

[18] J. PITRAT: A general game playing program. Artificial intelligence and heuristic programming. Edited by Findler and Meltzer, Edinburgh University Press 1971, p. 125-155

[19] B. FLAVIGNY: Un programme detecteur d'erreurs de logique dans les programmes. Revue Fran<?aise d'Automatique d'Informatique et de Recherche Operationelle. Informatique, Juillet 1975.

[20] A. ADAM, J.P. LAURENT: Mise en evidence de relations d'ordre dynamique entre les instructions d'un programme. Panorama de la Nouveaute Informatique en France, AFCET. Tome 1, p. 169-179.

[21] A. ADAM: Gadget: un programme de generation automatique de programmes sur les graphes et les ensembles. These de 3 Cycle, Paris VI, 2-4-73.

[22] P. GLOESS: Gener: un generateur de programmes, These de 3 Cycle, Paris VI, 19-10-76.

[23] F. GARIJO: Sudi: Un programme qui verifie des problemes et ecrit des programmes optimises. Panorama de la Nouveaute Informatique en France. Tome 1, p. 67-75-

[2k] M. MASSON: Un programme qui genere des programmes. Panorama de la Nouveaute Informatique en France, AFCET 1976, Tome 1, p. 381-390.

[25] J-L. LAURIERE: Un langage et un programme pour enoncer et resoudre des problemes combinatoires. These d'Etat, Paris VI, 21-5-76.

[26] G. SARAH: Un systeme de comprehension du langage naturel. Panorama de la Nouvearute Informatique en France, AFCET 1976, Tome 1, p 269-281.

[21] M.C. LOYO VARELA: Une experience informatique de comprehension du langage naturel. These de 3° Cycle, Paris VI, 8-I1-76.

[28] F. ROUSSELOT: Simulation d'un robot qui comprend et execute des ordres donnes en franijais. These de 3° Cycle, Paris VI, 13-6-75.

[29] M.F. VERDEJO: Etude du langage naturel. Simulation d'un robot capable de mener un dialogue en espagnol. These de 3 Cycle, Paris VI, 30-10-75-

[30] M. ROUSSEAU: Resolution automatique d'exercices d'electricite poses en francais. Revue franchise d'automatique d'informatique et de Recherche Operationelle. Informatique, Juillet 1975.

[31] D. COULON, D. KAYSER: Grammatical inference on natural language samples in the framework of C.A.I. Computer oriented learning processes. Edited by Simon, Nato advanced study institute series - Noordhoff 1976, p. 193-205.

For the other contributions, research reports can be obtained by writing to the respective projects, or to the panelists for that country.

PANELISTS

Affiliations of panelists:

J. Pitrat	University Pierre et Marie Curie (Paris VI) (France)
E. Sandewall	Linkoeping University, (Sweden)
W. Bibel	Technische Universitaet Muenchen (Munich, Federal Republic of Germany)
G. Huet	Institut de Recherche d'Informatique et d'Automatique (France)
H.-H. Nagel	Institut fuer Informatik, Universitaet Hamburg (Federal Republic of Germany)
M. Somalvico	Istituto de Elettrotecnica ed Elletronica, Politecnico de Milano (Italy)

Authors of the sections of this paper:

Introduction and post-editing	Sandewall
France	
Luminy Vincennes IRIA	Huet
Paris VI	Pitrat
Germany	Bibel and Nagel jointly
Italy	
Sweden	Sandewall

(Continued ->)