GARI : A PROBLEM SOLVER THAT PLANS HOW TO MACHINE MECHANICAL PARTS

Yannick DESCOTTE and Jean-Claude LATOMBE

I.M.A.G.
B.P. 53X - 38041 - GRENOBLE Cedex - FRANCE

ABSTRACT

In this paper, we describe a plan generator for planning the sequence of machining cuts of mechanical parts. This system, named GARI, makes use of weighted pieces of advice representing the expertise of human specialists. The complexity of the application domain guided us toward the implementation of a rather sophisticated control structure including hypothesis generation, fact deduction, and conflict resolution. Experiments with GARI suggest that more research effort should be devoted to abstract representation of control as opposed to the exhaustive representations defended by several authors during the last few years.

KEY-WORDS

Plan generation, expert systems, production rules, applications of AI, machining plans, conflict resolution.

I INTRODUCTION

In this paper, we describe a plan generator named GARI. This system makes use of an expert knowledge base for automating the planning of the machining sequence of mechanical parts.

Planning the machining process is regarded in the context of CAD/CAM as an important, but difficult problem. The main obstacle comes from the fact that producing a machining plan requires the application of a large amount of rather subjective and specialized knowledge. This knowledge is better described as many differently weighted pieces of advice than as a small set of constraints to be satisfied. Compromises are often necessary.

GARI makes use of such pieces of advice to generate plans for machining parts. The complexity of the application domain guided us toward the implementation of a rather sophisticated control structure including hypothesis generation, fact deduction, and conflict resolution.

During the last few years, several authors have advocated explicit representation of control for improving the behavior of problem-solvers (e.g.:/1/, /2/,/3/,/A/,/5/). We first implemented and experimented with such a representation drawn from the one used in TROPIC /A/. However, it quickly turned out to be inadequate : an intractable quantity of inferences and facts had to be remembered. Then, we moved toward an abstract, but easier to handle, representation of control, which experiments have proved to be more appropriate to the problem at hand.

We believe that research on the abstract representation of control will be an important topic in AI during the next few years, in order to solve really complex problems requiring a large number of inferences.

GARI is written in the MACLISP language and operates on the HB-68 computer. We have run it on industrial mechanical parts of considerable complexity.

II PROCESS PLANNING

The purpose of process planning is to produce a plan for machining a part, given its drawing. The plan should specify the machining cuts to be executed, their order, the machine tools to be used, the surfaces by which to body is clamped and located during each cut, etc.

Such planning involves taking into account both technological rules and economic considerations, for example :

- If a hole H2 opens into an other hole HI, then one is recommended to machine HI before H2 in order to avoid risks of damaging the drill,

- It is advantageous to execute several cuts on the same machine with the same fixing for reducing the time spent in setting up the work on the machine (we then say that these cuts are grouped into the same "phase").

There is a great variety of such technological rules and economic considerations. Furthermore, they are not absolute. They are more preferences, between which compromises may be necessary. In addition,they may differ somewhat from one company to another. In fact, they represent the experience and the know-how of engineers.

An immediate consequence is that the planning of machining sequences definitely does not have a unique solution.

III OVERVIEW OF GARI

GARI is structured like an expert system /6/. It consists of a specialized knowledge base and a more general purpose problem-solver.

Knowledge is represented by production rules. The left-hand side of a rule is a set (conjunction and/or disjunction) of conditions about the mechanical part, the available machines, and/or the machining plan. Items in the right-hand side are called pieces of advice. They are sets of facts representing technological or economic preferences. Each piece of advice is weighted according to the importance of its satisfaction.

GARI generates a machining plan from a model of the part. This model describes the part in terms of entities, such as holes, grooves, notches and faces, and it includes both geometrical and technological data. It a priori determines the set of potential cuts. Indeed, each entity may require a maximum of two cuts : a roughing cut and a finishing cut.

Plan generation proceeds through successive refinements. Each iteration produces new assertions that apply to the set of potential cuts, for example : cut A is executed before cut B, cuts C and D are grouped in the same phase, cut E is performed on a grinding or a milling machine. These assertions, each of which constraints the solution a little more, are selected according to the pieces of advice that are provided by active instances of rules (i.e. rules the left-hand sides of which are currently true). At any moment, the assertions that have been produced so far define a finite set of plans. Indeed, they imply a partial order on the set of potential cuts and they specify incompletely the characteristics of each cut. However it may happen that some assertions are contradictory and so determine an empty set of plans.

A contradiction implies that some pieces of advice are conflicting. To solve it, GARI selects and rejects a piece of advice. Facts inferred from it are no longer assumed. Then, the iterative planning process is resumed. If it later becomes useful, GARI can reconsider the solution to a conflict, in order to reapply a piece of advice that has been previously discarded.

When the process of successive refinements is complete, the system has reached a state where each of the pieces of advice provided by all the active rules is either satisfied, or contradicted (due to conflicts) by the current set of assertions. Then, there may exist rules the left-hand side conditions of which are neither satisfied nor contradicted. For each particular plan P determined by the current assertions, one or several of these rules may become active and may provide pieces of advice that contradicts P. Therefore, GARI generates hypothetical assertions in order to make these rules definitely inapplicable. Of course, these new assertions may in turn produce new conflicts.

Assertions are selected and conflicts are solved so as to minimize the maximum weight of those pieces of advice which are finally contradicted.

The final set of assertions produced by GARI does not usually determine a unique plan. Remaining choices can often be solved in a better way at execution time when the machine utilizations are known.

IV RELATIONS TO OTHER WORKS

Other programs exist in the field of process planning for mechanical parts/7/. Most of them attempt to automate "low level" tasks such as providing a detailed analysis of a particular operation (for example : machining a hole), computing machining times, etc. These programs make no use of AI methods and may be regarded as complementary to GARI. Other programs operate only on parts belonging to a given technological family (typically turned parts). At a more fundamental level, there exists an attempt to apply elements of fuzzy set theory /8/. However, we do not believe that a mathematical formulation of the process planning problem is appropriate. On the other hand, we found no substantial AI work on this problem.

GARI has some evident analogies with plan generators such as STRIPS /9/. However, both the problem to be solved and the methods used to solve it are different with some regards :

(1) GARI knows beforehand the set of potential actions (cuts) to be executed : the problem is mainly to order them and to specify how they should be executed.
(2) Properly speaking, GARI does not have an explicit model of these actions, with preconditions, delete and add lists, like most of the plan generators. In fact, some of these model elements exist implicitly.
(3) Most of the existing plan generators are general purpose problem-solvers. GARI is specialized in the field of process planning and makes use of an expert knowledge base.

There exists a rather strong relation (at least at the conceptual level) between the pieces of advice applied by GARI and the critics applied by NOAH /10/. Although NOAH critics are general purpose ones, the use of domain specific ones was suggested by Sacerdoti. On another hand, GARI does not currently perform hierarchical planning like NOAH, but we have this type of planning in mind.

There also exist analogies between GARI and traditional rule-based expert systems /6/. The main difference concerns control structure, which is substantially more complex in GARI than it is in typical diagnosis expert systems like MYCIN /11/.

The conflict resolution procedure derived initially from the failure processing procedure implemented in TROPIC /A/. It is now very different, partly due to reasons presented in the introduction of this paper. Another major reason is that TROPIC, like the TMS of Doyle /3/, faced only failures produced by the contradiction of constraints, the satisfaction of which was imperative. In contrast, GARI faces conflicts between pieces of advice, which may be regarded as weak constraints, so that it is valid to make compromises between them.

V PART DESCRIPTION

The shape of a part is described to GARI as a basic volume (e.g. : a parallelepiped, a cylinder) together with a specification of instances of

"entities" of the type used by engineers in shape description : holes, grooves, notches, bores, faces, etc. Dimensions and tolerances are input as they appear on the drawing, and thus require no computation in absolute coordinates. Technological information existing on the drawing (e.g. surface quality) is also given to GARI.

Example :

The countersunk hole H of figure 1 is described to GARI as follows :

```
(H (type countersunk-hole)
   (starting-from F1)
   (opening-on F3)
   (diameter 6)
   (countersink-diameter 12)
   (surface-quality 7))

(distance H F2 19)
(countersink-depth H F1 3 +-80)
(perpendicularity H F3 +-50)
```

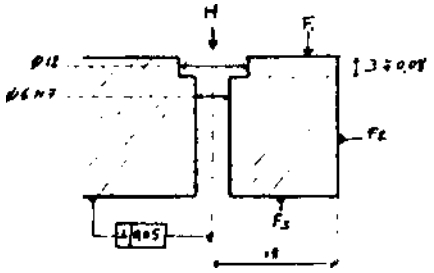

Figure 1

The notion of entity is important because it reflects a basic concept frequently used by process planning specialists. Not only does it make part description easier but, in addition, the statement of rules is simplified by the semantic value that is implicitly attached to each type of entity.

In its current implementation, GARI can make use of about twenty different types of entities.

VI MACHINE DESCRIPTION

Each available machine is described to GARI by a name and by properties, which are regarded as important by the users. For instance, some properties may be the type of the machine (milling machine, drilling machine, etc.), its precision, dimension attributes such as the distance between axes, etc.

Example :

```
(CL1 (type chuck-lathe) (chuck-diameter 2.0)).
```

VII RULES

The form of each rule used by GARI is the production :

conditions ■■> pieces of advice

Each piece of advice is weighted by an integer between 1 and 10 that represents the importance attached to its satisfaction (10 - most important). It consists of a list of assertions to be added to the current solution when it is applied.

Describing all the primitives that are useful for writing rules would be fastidious here. We believe it is more appropriate to illustrate them on some sample rules.

Atoms prefixed by &, &&, $, $$, and ::: are variables representing entities, sets of entities, machining cuts, sets of machining cuts and sets of machines respectively.

```
- (>= (surface-quality Ax) 6.3)
  ==>
  (9 ("roughing-cut Ax))
     If the surface quality of an entity Ax is
     higher (poorer) than 6.3, then one is advised
     (with a weight equal to 9) to avoid a roughing
     cut for Ax.
     /Note : GARI interprets such a piece of advice
     by grouping the roughing cut and the finishing
     cut of Ax in the same "operation" (set of cuts
     to be realized simultaneously). This simple
     trick permits the system to work on a constant
     set of cuts./

- (geometrical-eonstraint-between Ax Ay)
  (=1 (finishing-cut Ax) (finishing-cut Ay))
  ==>
  (A (roughing-cut Ax) (roughing-cut Ay))
     If Ax and Ay are entities linked by an input
     geometrical constraint and if their finishing
     cuts are not executed in the same operation
     (i.e. simultaneously), then one is advised (4)
     to execute roughing cuts for both entities.
     /Note : the left-hand side of this rule depends
     on the plan to be generated./

- (<= (surface-quality Ax) 3.2)
  (>- (extra-thickness Ay) 3.0)
  (supported-by (roughing-cut Ay) Ax)

  (8 (> (finishing-cut Ax) (roughing-cut Ay)))
     If the surface quality of an entity Ax is lower
     (better) than 3.2, and if the extra-thickness
     of an entity &y is greater than 3.0, and if the
     part is supported by &x during the roughing
     cut of &y, then one is advised (8) to perform
     the finishing cut of Ax after the roughing
     cut of &y.

- (is-a $x finishing-cut)
  (machine $x grinding-raachine)
  ("machine $y grinding-machine)
  ==>
  (10 (> $x $y))
     If finishing cut $x is executed on a grinding
     machine, and if cut $y is executed on another
     kind of machine tool, then one is advised (10)
     to perform $x after $y.

- (is-a Ax bore)
  (is-a Ay bore)
  (coaxial &x &y)
  ==>
  (6 (=2 (finishing-cut Ax) (finishing-cut Ay))
       (machine (finishing-cut Ax) lathe))
```

If &x and &y are two bores linked by a coaxiality constraint, then one is advised (6) to execute their finishing cuts in the same phase (i.e. with the same fixing on the same machine) on a lathe.

- (is-a &x hole)
  (is-a &y hole)
  (open-into &x &y)
  ("open-into &y &x)
  ==>
  (9 (> (roughing-cut &x) (roughing-cut &y)))
  If &x is a hole that opens into a hole &y, and if &y does not open into &x, then one is advised (9) to execute the roughing cut of &x after the roughing cut of &y.

- (is-a &x bore)
  (< (surface-quality &x) 6.0)
  ("empty (machine-y cylinder-grinding-machine
  (<= (surface-quality machine-y) (surface-quality &x))))
  ==>
  (10 (machine (finishing-cut &x) machine-y))
  If the surface quality of a bore &x is lower (better) than 6.0, and if a cylinder grinding machine able to provide this quality is available, then one is advised (10) to execute the finishing cut of &x on such a machine.

A rule instance (that is a rule in which constants have been substituted for variables) is said to be "active" when the conditions in its left-hand side are satisfied by the part model, the machine description and the current set of assertions generated by GARI. It is said to be "inapplicable" when one of these conditions is contradicted. In all other cases, it is said to be "pending".

Presently, GARI runs with a knowledge base containing more than fifty rules, most of them providing several pieces of advice. For a part of moderate complexity, it is often the case that more than ten instances of the same rule become active.

In the following, the word "rule" will also apply to a rule instance.

VIII GENERATING ASSERTIONS

GARI iteratively generates assertions by applying pieces of advice of decreasing weight. More precisely, it proceeds according to the two following steps :

(1) While active rules provide pieces of advice, which were not discarded when solving previous conflicts, GARI applies one of these pieces of advice drawn from those with the highest weight.
(2) While pending rules provide pieces of advice that are not satisfied by the current solution, the system generates hypotheses with the goal of making these rules inapplicable. These hypotheses are regarded as implicit pieces of advice of weight 0. If one of them activates a rule providing a piece of advice that is not currently satisfied, then step (I) is executed again.

Each time an explicit or implicit piece of advice is applied, GARI checks that all assertions produced so far are consistent. If a contradiction is detected, the procedure for solving it is immediately executed.

Detecting inconsistencies may require complex deductions that should not be repeated. Therefore, after applying each piece of advice, GARI performs all new deductions it can and records the concluded facts in the assertion data base.

Suppose for instance that the following four assertions have been generated :
- (-2 (finishing-cut F2) (finishing-cut FA))
  /The finishing cuts of F2 and FA are grouped into the same phase./
- (-2 (finishing-cut Fl) (finishing-cut F5))
- (> (finishing-cut F2) (finishing-cut Fl))
  /The finishing cut of F2 is executed after the finishing cut of Fl./
- (=2 (finishing-cut F2) (finishing-cut Fl))
  /The finishing cuts of F2 and Fl are not grouped into the same phase./

GARI can deduce the following fact (among other conclusions) :
- (> (finishing-cut FA) (finishing-cut F5))
  This deduction is based on the meaning attached to the notion of phase (set of cuts to be executed with the same fixing on the same machine).

Then, if the assertion
- (> (finishing-cut F5) (finishing-cut F4))
is generated, a contradiction can immediately be detected.

Many deductions performed by GARI are based on more complex reasoning than the above one. Indeed, there may exist disjunctive assertions such as :
- (machine (finishing-cut Bl) lathe boring-machine)
  /The machine used to perform the finishing cut of Bl is either a lathe, or a boring-machine./

Thus, recording conclusions results in substantial time saving. Of course, it requires more memory space, but we have implemented a representation imbedding assertions into a network structure. This representation subtantially reduces the necessary space.

/Note ; The meaning attached to such notions as operation, phase, machine, roughing cut, etc., which is necessary for performing deductions, is proceduraly imbedded into the programs of GARI. It could also have been represented by the means of inference rules. Although the solution we choose is less general, it makes the system more efficient. The complexity of the deductions to be made is often such that we believe that a general purpose solution would have been unrealistic./

If, after applying a piece of advice, no contradiction is detected, a program for activating rules is called. This program determines all new active rules given the assertions that were just generated. It records its work into activation trees /12/, that make updating easier when a conflict is processed. The activation program makes use of the recorded deductions.

IX CONFLICT RESOLUTION

When a contradiction among assertions is detected, it is immediately processed. Basically, the system discards a previously applied piece of advice and updates the current solution accordingly.

The implemented procedure is based on a few simple concepts, which subsequent experiments proved to be rather efficient with respect to the problem at hand :

a) GAR I always rejects the piece of advice the most recently applied among those which have the minimum weight.

For example, if the following pieces of advice
AI A2 A3 AA A5 A6
with the following weights
10 7 9 7 10 9
were applied successively when a contradiction is detected, GARI discards AA.

If it is the first contradiction, this piece of advice is necessarily responsable of the contradictioa Indeed, assertions are generated according to pieces of advice of decreasing weight. Thus, assertions produced by the application of AA are necessary to the activation of those rules which provided A5 and A6. If this was not the case, those rules would have become active before the application of AA. So, A5 and A6 would have been used before AA.

b) Rejecting a piece of advice must be accompanied by updating the current solution. Thus, assertions generated when this piece of advice was applied must be removed. This usually leads to disactivate some rules. If these rules provide pieces of advice that were previously applied, the corresponding assertions must be removed in turn. And so on..

According to the reasoning of a), GARI is led to remove all assertions produced after the application of the piece of advice which is discarded. In the previous example, this is equivalent to remove the assertions produced by the application of AA, A5 and A6 (including the deduced facts).

c) A contradiction is always due to a conflicting combination of pieces of advice. Thus, if a new contradiction occurs later, it may become useful to reintroduce a previously rejected piece of advice. However, GARI must also solve the first contradiction in a different manner in order to avoid an infinite loop of successive failures.

For this purpose, the system proceeds as follows : Each time it discards a piece of advice X, it generates assertions corresponding to the application of the implicit piece of advice ~X (not X). ~X is then regarded as a real piece of advice. GARI gives it a weight w equal to the minimum weight of those pieces of advice which were applied after X. This weight is justified as follows : if we decided to maintain X in the same conditions than it was applied (that is without modifying the pieces of advice applied before X), then we would meet again a contradiction the solution of which would "cost" at least the discard of a piece of advice having weight w. In the example of a), this corresponds to replace AA by "A4 with weight 9.

If the rejected piece of advice X is the most recently applied one, then "X is introduced with weight 11. Indeed, applying X without modifying previously applied pieces of advice would lead to a definitive contradiction.

If no new contradiction is detected immediately after having introduced ~X, the program for activating rules is executed. Then, the normal process of generating assertions is resumed.

The actual procedure implemented in GARI is somewhat more complex than it is shown above. However, the extra-complexity currently is based more on intricate tricks, which are evolving quickly, than on clear principles.

X A SIMPLE EXAMPLE

The following example illustrates the inputs and outputs of GARI. The part to machine is defined by the drawing of figure 2. It is initially given as a rough casting and only those surfaces represented by thick lines must be machined.

This part was input as a rectangular parallelepiped with holes and notches. We also described to GARI a classical set of machines made of a parallel lathe, a milling machine, a drilling machine, a boring machine, a surface planing machine, a surface grinding machine and a cylinder grinding machine.

The solution produced by GARI consisted of a set of machining plans characterized by the following properties :

- The first phase of each plan consists of executing the finishing cut of FZM on the milling machine, the part being supported by face FZP.

- Then, the part being supported by FZM, the following three operations should be executed in sequence :
  - finishing cut of FZP on the milling machine,
  " roughing cuts of notches N21 and N22 on the milling machine,

  :: finishing cuts of N2) and N22 on the surface planing machine or on the milling machine.

No information was provided on how to group these operations into one or several phases.

- The last phase of each plan is executed on the drilling machine, the part being supported by face FZM. It consists of two operations, which can be executed in any order :
  :: finishing cuts of holes HII, H12, and H2,
  ;: finishing cuts of all other holes.

This example is very simple and only five pieces of advice were to be discarded. One recommended a roughing cut of face FZM, because it is the first entity to be machined. However, it contradicted a more important piece of advice related to the poor surface quality required for FZM. Other discarded

pieces of advice recommended the execution of the roughing and finishing cuts of the notches N21 and N22 with the part being supported by face FZP (in order to get a better precision). But this was not compatible with the available tool orientations of the milling and surface planing machines.

XI DISCUSSION AND PERSPECTIVES

An implementation of GARI written in the MACLISP language operates on a HB-68 computer under the MULT1CS system. We have run it on industrial mechanical parts of considerable complexity. The description of some of them included more than 100 entities (cf. Section V.). Almost all generated machining plans were acceptable. The system is now being used experimentally by machining planning engineers.

Although it may not clearly appear in this paper, large efforts were expended for representing machining expert knowledge. It required a detailed analysis of the machining planning problem, acquired by working with engineers. Significant efforts were also devoted to fact deduction and to geometrical modeling.

Our own experimentation proved the global adequacy of the methods described here. However, current experimentation by experts is resulting in the rule data base being augmented. A rough evaluation showed that the number of rules should grow from the current 50 to about 100 rules. This growth already pointed out new problems

A larger number of rules usually results in more conflicts among pieces of advice. Current conflict resolution is based on simple concepts which are likely to become insufficient soon. In fact, as we already said, we first implemented a strongly dependency-directed backtracking method, which proved to be inadequate because of the large amount of dependencies to be recorded. This is due to the fact that many assertions may be derived by different inferences.

we turned to an abstract representation of control where each conflict is remembered by applying "X (not the piece of advice that is discarded), to which an appropriate weight is associated. Clearly, this is too naive. There should exist a better compromise between this representation and the exhaustive recording of dependencies. We believe that this problem is representative of many complex applications, so that it is likely to become an important topic in AI during the next few years.

Augmenting the rule data base also showed that many rules have exceptions. At the present time, each exception is represented as another rule. For instance,

C1, C2, ... Cn, Cn+1 ==> PA'

may represent an exception to rule

C1, C2, ... Cn —> PA.

This representation leads to uninteresting conflicts that may cause a significant waste of time. It also makes difficult for the user to get a global view of the rules. We have in mind new methods enabling the system to work directly on rules of the form :

C1, C2, ..., Cn —> PA

Exception : Cn+1 ==> PA[1].

Another interesting perspective concerns hierarchical planning /10/. To date, GARI has no global view of the plans it generates. In contrast, when human engineers are confronted to a part above a certain level of complexity, they often attempt to work first on a simplified representation of the part, by eliminating "unimportant" entities. Therefore, we think about introducing new rules describing how to simplify complex parts, so that the system can perform reasoning at various levels of abstraction.

ACKNOWLEDGEMENTS

This work has been done under contract and in collaboration with ADEPA (Agence Nationale pour le Developpement de la Production Automatised). We thank Messrs. FERNET, MERLIN, MOLLON, of ADEPA, and Mr BOURDET, of ENSET, for their assistance in establishing the knowledge base of GARI.

REFERENCES

/1/ P.J. HAYES (1975)
"A representation for robot plans",
IJCAI, Tbilisi, 1975.

/2/ R.M. STALLMAN & G.J. SUSSMAN (1977)
"Forvard reasoning and dependency-directed backtracking in a system for computer aided circuit analysis",
Artificial Intelligence, vol. 9, 1977.

/3/ J. DOYLE (1977)
"Truth maintenance systems for problem solving",
S.M. Thesis, MIT, 1977.

/4/ J.C. LATOMBE (1979)
"Failure processing in a system for designing complex assemblies",
IJCAI, Tokyo, 1979.

/5/ J. DOYLE (1980)
"A model for deliberation, action, and introspection",
MIT, AI TR-581, 1980.

/5/ E.A. FEIGENBAUM (1977)
"The art of artificial intelligence : I. Themes and case studies of knowledge engineering",
IJCAI, Cambridge, 1977.

/7/ R. WEILL (1973)
"Review of some new techniques for the optimisation of manufacturing systems",
Euromachines, 1973.

/8/ BEN SALEM (1976) (in French)
"Sur l'extension de la theorie des sous-ensembles flous a 1'automatique industrielle (mise au point d'une nouvelle methode d[fe]tude des gammes d'usinage)",
These de 3eme cycle, Paris, 1976.

/9/ R.E. FIKES & N.J. NILSSON (1971)
"STRIPS : a new approach to the application of theorem proving to problem solving",
Artificial Intelligence, vol. 2, 1971.

/10/ E.D. SACERDOTI (1975)
"A structure for plans and behaviors",
SRI, AIC TN-109, 1975.

/11/ E.H. SHORTLIFFE (1976)
"*MYCIN : computer-based medical consultations",
New-York, American Elsevier, 1976.

/12/ J.C. LATOMBE (1977) (in French)
"Une application de 1'intelligence artificielle a la conception assistee par ordinateur (TROPIC)",
These d'Etat, Grenoble, 1977.

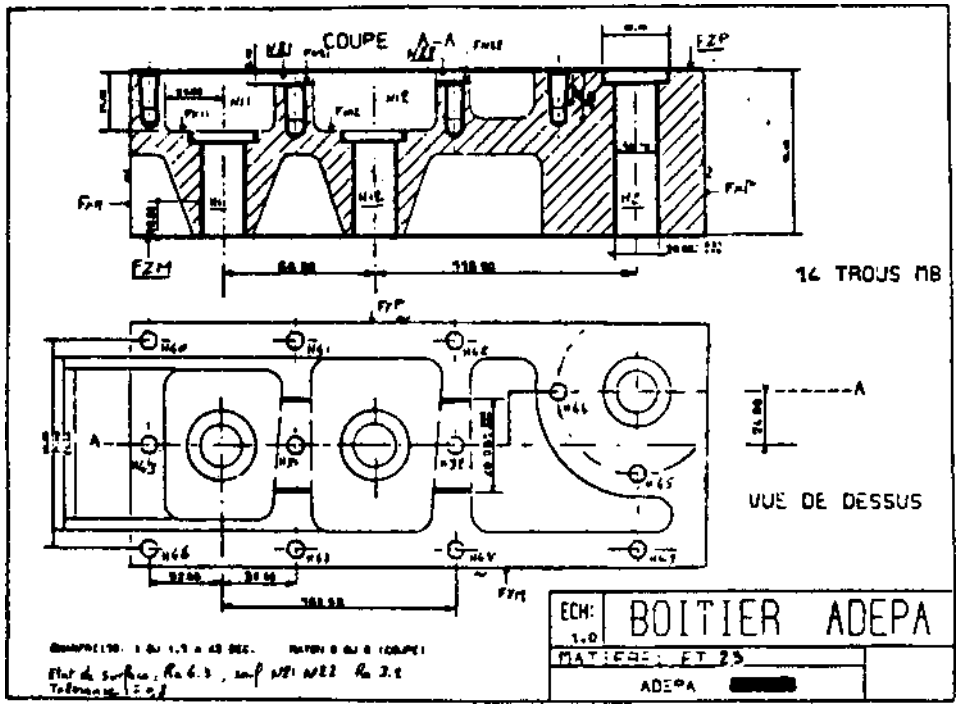

Figure 2