How to use MALICE, CAIA's solver

Jacques Pitrat

When I mention “the paper”, this is “A Step toward an Artificial Artificial Intelligence Scientist” available in this site, and when I mention “the book”, this is “Artificial Beings. The conscience of a conscious machine”, 2009, ISTE and Wiley. Both are using results found by CAIA during one of its lives at the end of 2005.Many changes has been made to CAIA during the last years for giving it the possibility to solve arithmetic problems. Thus, it is possible that the trees or the execution times of the present version are not exactly the same as those found several years ago.

1. Preparing the program

Quick commands to run:

decompress the zipped archive with

```
bunzip2 malice-2009.tar.bz2
```

then extract the files

```
tar -xvf malice-2009.tar
```

go into the newly made directory

```
cd malice-2009
```

compile all the source files and link them

```
gcc -O2 *.c -rdynamic -ldl -o malice
```

This should take several minutes, since Malice contains several hundred thousands of C code (generated by itself). You should not have any warnings or error messages from the `gcc` compiler. The `-ldl` library and the `-rdynamic` option are essential, since Malice may generate and then dynamically link some code.

you can now start Malice with

```
./malice
```

you should start Malice in the directory containing its source files and the dx.h dy dz files.

The program is running with the operating system Linux (any not too old Linux system with a 2.4 or 2.6 kernel and gcc 3.x or gcc 4.x should be ok, provided it has at least 1Gb of RAM and 4Gb of free disk space ; you'll need the usual packages required for software development in C, notably the gcc compiler and standard include files) and it works only with little-endian (or bi-endian) computers, such as x86 processors (either in 32 bits or in 64 bits mode, ie Pentium, Core2, AMD64 etc..). Particularly it cannot be executed on computers with a big-endian processor such as Motorola, or the first Power PC and Sparc because they could not read the dy and dz files.

The files dy, dz (binary files containing the persistent store required by Malice), dx.h, and all the *.c source files should be in the current directory.

The compressed file also includes the GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007.

2. Solving a problem

In all the situations, when one types 'F', one returns to the choice of a family of problems, when one types 'P', one returns to the choice of a particular problem in the lastly defined family, and when one types '.', one stops the session.

The program indicates first the name of the families of problems. One must type one of them. If one adds '?' at the end, the formulation of this problem is displayed.

When there is more than one problem in this family, the names of its problems are displayed. Thus, one types one of these names. If it is followed by '?', the particular data of this problem are displayed. When the name of a problem begins with MAL, it has been created by CAIA.

Most problems can be solved by three methods. Depending on the problems, one of these methods may be faster than the others, but the tree generated with the intelligent method is always much smaller.

1. The intelligent method, where MALICE finds a solution with a very small tree, similar to the solutions created by humans.

2. The combinatorial method where MALICE begins to write a combinatorial program, compiles it, loads it and executes it.

3. The improved combinatorial method where MALICE begins with the intelligent method, reduces the size of the search space, finds new useful constraints until it can do nothing other than backtracking. In that situation, it writes a combinatorial program, and executes it in the same way as in the preceding paragraph.

The program asks the user what kind of solution he prefers. If he wants the intelligent method, he types the entry key. For the purely combinatorial method, he types 'D', and 'C' for the combinatorial method after an intelligent phase. When a combinatorial program is created, its name is COMB0.c, and its data are in db.h. Typing 'D' can be very fast when the search space is small, only a few million leaves. For larger spaces, it is usually faster to type 'C'. For a few problems, such as &ARROWS, &LOGIGRAPH, or &AUTOREF, the intelligent method may be faster than when one types 'C' or 'D'. The tree developed with the intelligent method is always very much smaller than with 'C', which is itself smaller than with 'D'.

For some problems, CAIA is not yet able to write a combinatorial program, essentially those where is has to find correspondences which are multivalued functions (for instance &EULER and &ROOKSTOUR) and those where there is an imbrication of correspondences, for example when there is a constraint such as F{F{3}}=5 (this happens in &QGROUP). If this is the case, the question is not asked, and only the intelligent method is used.

Then MALICE asks for the maximum time allowed to the execution. The default value is 60 seconds, and in this case it is sufficient to use the return key. If one wants to indicate another limit, one types 'T' followed by the number of seconds. If the allocated time was not sufficient to complete the resolution, MALICE indicates the percentage of the tree that has been developed; then, it is possible to allow it to spend more time on the problem.

Finally, one types 'E' for launching the execution.

The idea of the intelligent method was introduced by Jean-Louis Laurière in ALICE; many rules can reduce the size of the search space. In that way, a system performs a meta-combinatorial search rather than a combinatorial search: it considers the possible rules rather than the possible values of a variable. It generates solutions similar to those found by good human problem solvers, this is why I call this approach the intelligent method. This important point is developed in the paper p.19-20.

CAIA can explain its solutions, that is it indicates the deductions that it made for restricting the search space. It can also meta-explain a solution, that is it can indicate why it has considered these deductions. As it does it for its own needs, this is not always clear for a human being. Thus it would be necessary to modify drastically the presentation of these results so that a human user could understand them easily. As this modification has not been made, I do not include the explanations and meta-explanations in the results.

3. Printing the solution

First, MALICE prints the data of the problem, using the name of the variables which appear in the formulation of the problem. Their meaning is indicated in the section describing the various families. Then it prints the initial algebraic constraints of the problem, and the solutions that it has found. It makes a report on its resolution, possibly indicating only a partial success. Finally it prints the tree that it has generated with the intelligent method; if it has generated a program after an intelligent period, it only prints the partial tree such it was at the end of this intelligent search. In this tree a sequence of '#' indicates a solution, while '*' is for a failure, or possibly for an undeveloped node if the search has not been completed.

Here is an example of the output of the resolution by MALICE of the problem DONALD of the family &KRYPT. We first start the program from a shell, ie inside a terminal: `./malice`.

```
YOU CAN ALWAYS TYPE F FOR CHOOSING ANOTHER FAMILY OR P FOR CHOOSING ANOTHER
PROBLEM OR . FOR STOPPING THE SESSION.
IF YOU TYPE ? AFTER THE NAME OF A FAMILY, I ALSO LIST ITS FORMULATION.
IF YOU TYPE ? AFTER THE NAME OF A PROBLEM, I ALSO LIST ITS DATA.


CHOOSE A PROBLEM FAMILY AMONG: [&EULER,&KRYPT,&INSANITY,&KRYPTUN,&KRMUL,
&SUDOKU,&ROOKSTOUR,&ARROWS,&GOLOMBRULER,&LANGFORDSNUMBER,&AUTOREF,&KAKURO,
&LOGIGRAPH,&QGROUP,&CUBETOTAL,&ALLINTERVAL,&STEINER,&SOCCERTABLE]
! &KRYPT
CHOOSE A PARTICULAR PROBLEM OF THE &KRYPT FAMILY AMONG: [`SQUARE`,`FORTY`,
`SOMME`,`DONALD`,`MONEY`,`WASTER`,`SUN`,`ADAM`,`ENVOIE`,`VINGT`,`ABA`,
`FOOT`,`CUATRO`,`YOUNG`,`EXAM`,`TWENTY`,`TRENTE`,`MAL1`,`MAL2`,`MAL3`,
`MAL17`,`MAL4`,`MAL18`,`MAL21`,`MAL7`,`MAL15`,`MAL40`,`MAL31`,`MAL29`,
`MAL34`,`MAL22`,`MAL33`,`MAL36`,`MAL25`,`MAL27`,`MAL37`,`MAL38`,`MAL32`,
`MAL8`,`MAL26`,`MAL28`,`MAL30`,`MAL14`,`MAL6`,`MAL9`,`MAL5`,`MAL10`,`MAL13`,
`MAL16`,`MAL12`,`MAL19`,`MAL39`,`MAL41`,`MAL20`,`MAL42`]
! DONALD
FOR USING A COMBINATORIAL PROGRAM, TYPE C (AFTER AN INTELLIGENT
REFORMULATION) OR D (WITHOUT IT).
?
IF MAX TIME IS NOT 60 SECONDS TYPE T FOLLOWED BY THE NUMBER OF SECONDS.
?
TYPE E FOR STARTING THE SOLVER OR F FOR CHOOSING ANOTHER FAMILY OR OR P FOR
CHOOSING ANOTHER PROBLEM OR . FOR STOPPING THE SESSION.
! E
I WILL EXECUTE &KRYPT DONALD
NC: 6
NL: 3
```

```
A
1  DONALD
2  GERALD
3  ROBERT

R{0}=0
R{6}=0
SUM[R{6},PRD[2,F{'D'}]]=SUM[PRD[10,R{5}],F{'T'}]
SUM[R{5},PRD[2,F{'L'}]]=SUM[PRD[10,R{4}],F{'R'}]
SUM[R{4},PRD[2,F{'A'}]]=SUM[PRD[10,R{3}],F{'E'}]
SUM[F{'R'},F{'N'},R{3}]=SUM[F{'B'},PRD[10,R{2}]]
SUM[R{2},F{'E'}]=PRD[10,R{1}]
SUM[F{'G'},F{'D'},R{1}]=SUM[F{'R'},PRD[10,R{0}]]
F{'R'}#0
F{'G'}#0
F{'D'}#0
THERE ARE 11 INITIAL ALGEBRAIC CONSTRAINTS.

SOLUTION 1
"R{0}"= "{0}R"
"R{1}"= "{1}R"
"R{2}"= "{1}R"
"R{3}"= "{0}R"
"R{4}"= "{1}R"
"R{5}"= "{1}R"
"R{6}"= "{0}R"
"F{'D'}"= "{5}F"
"F{'O'}"= "{2}F"
"F{'N'}"= "{6}F"
"F{'A'}"= "{4}F"
"F{'L'}"= "{8}F"
"F{'G'}"= "{1}F"
"F{'E'}"= "{9}F"
"F{'R'}"= "{7}F"
"F{'B'}"= "{3}F"
"F{'T'}"= "{0}F"
```

```
I GENERATED THE FOLLOWING TREE WITH THE INTELLIGENT METHOD:
1.R{0}=0 2.R{6}=0
 3.1.F{'E'}=0 4.R{4}=0 5.R{2}=0 6.R{1}=0 7.F{'A'}=5 8.R{3}=1 9.R{5}=0
  10.F{'R'}=6 11.F{'L'}=3 12.F{'T'}=8 13.F{'D'}=4 14.F{'G'}=2 15.F{'N'}=1*
 3.2.F{'E'}=9 4.R{4}=1 5.R{2}=1 6.R{1}=1 7.F{'A'}=4 8.R{3}=0 9.F{'T'}=0
  10.F{'D'}=5 11.R{5}=1 12.F{'R'}=7 13.F{'L'}=8 14.F{'G'}=1 15.F{'B'}=3
  16.F{'N'}=6 17.F{'O'}=2 ########

I USED MY EFFICIENT METHOD, I FOUND 1 SOLUTION IN 2 SECONDS FOR THE &KRYPT
  DONALD PROBLEM, AND I DEVELOPED A TREE WITH 2 LEAVES.
```

We have first the data, then the initial constraints as they are generated by MALICE using the data, the solution, and finally the tree generated by MALICE for finding the solution and proving that it is the only solution.

The tree has only two leaves; R{I} stands for the Ith carry. It is not very difficult to explain most of the steps, for instance to show that E can have only two values 0 and 9. If one considers E=0, it is very easy to find R4=0, R2=0, R1=0, A=5, and R3=1. On the contrary, the explanation contains many difficult deductions for proving that R5=0. Then, it is easy to show that there is a contradiction. A tree shows the sequence of steps taken by MALICE, but some steps are difficult to understand. It happened several times that I was surprised by the small size of the tree generated for solving a problem; I had to examine the explanation for understanding what clever deductions MALICE had made. For several years, students in AI at Paris 6 had to solve by themselves the preceding crypt-addition so that they understand this intelligent problem solving method; none of them has ever found a solution with less than five leaves.

4. The families of problems

I preferred to include the families of problems including many problems, that is only 18 families among the 178 families of CSP problems whose formulation has been given to MALICE.

4.1. &EULER

Circuit of a piece on a NxN board. The piece is a hopper (P,Q), if P=2 and Q=1, this is the chess Knight, if P=4 and Q=1, this is a Giraffe, a kind of fairy piece. The squares are numbered from 1 to 64. The 2 values of the multivalued function F{I} are the two neighbors of square I in the circuit.

4.2. &KRYPT

Crypt-addition. NC is the number of columns and NL the number of lines, the last one being the sum. The matrix A contains the letters, where '*' represents a blank. This formulation uses carries; F gives the value of the letters and R of the carries. MONEY, DONALD, and WASTER given p.242 of the book are among the problems of &KRYPT.

4.3. &INSANITY

This is the well studied Instant Insanity problem, where one has to stack four cubes so that the faces of the cubes on each column are the same. CUBE indicates the six colors of each face of the four cubes, and OPP indicates the face opposed to each face. The place of each cube in the solution is defined by the names of its forward face FA and of its left face GA. The last constraint of the formulation eliminates symmetries.

4.4.&KRYPTUN

This formulation of crypt-additions does not use carries. The data are the same as with &KRYPT. The problem ABC+... given p.40 of the paper is MAL38 where the letters have been modified. The other problem given this page, ABCDE+..., has finally been eliminated later by CAIA, too many solutions.

4.5. &KRMUL

Crypt-multiplications. NC is the number of columns and A the three lines matrix containing the letters. The crypt-multiplications given p.190 of the book are MAL16, LYNDON, TRY, and HOPE.

4.6. &SUDOKU

The squares are numbered from 1 to 81. The sequence P gives the initial value of the squares, '0' standing for Blank. The function NBVAL[I;X1,X2,....,Xn] is used in the constraints; its value is the number of expressions Xi whose value is I.

4.7. &ROOKSTOUR

One must find a circuit of a rook on a square board. Some squares of the board are black which means that they are forbidden. N is the size of the board, and NN the number of white squares. The squares are numbered from 1 to NN. The vector A gives the number of the squares, 0 corresponding to a black square; it begins at the bottom left and ends at the top right; it indicates where are the black squares. B gives the number of each white square if the board was numbered from 0 to NxN-1: B(I) is the number of the Ith white square. An example of this problem was given in the newspaper *Le Monde*. The problem given in the paper p.40 is MAL73, and the one given in the book p.243 is MALICEJUN1. The multivalued function F{I} gives the value of its two neighbors; I and the values of its neighbors are numbered according to their rank among the white squares, that is they are included between 1 and NN.

4.8. ARROWS

A problem of this family was published in the newspaper *Le Monde*. In a solution, the content of each square must be the cardinal of the set of the values of the squares pointed by the arrow which is in this square. A problem is given in the paper p.36 (MAL3), and another in the book p.244 (MAL1). I have made a mistake in the picture in the paper, the middle arrow of the second line must point to the right and not to the top. Curiously enough, the solution is correct with both versions, both require a 3 in this square!

M indicates the number of columns, and N the number of lines. Vector H indicates the orientation of the arrows, beginning by the bottom line and going from left to right. 1 is for an arrow towards the top, 2 to the right, 3 to the bottom , and 4 to the left.

4.9. GOLOMBRULER

This is problem 6 in the CSPLib (http://www.csplib.org/). A Golomb ruler is a set of M integers Ai starting with 0 such as Ai<Aj if i<j, and the m(m-1)/2 differences Aj-Ai, i<j, are distinct. M is m and P is the value of the larger number Am. The name of a problem is defined by the value of M in French followed by the value of P: for problem DIX55, we have M=10 and P=55.

4.10. LANGFORDSNUMBER

This is problem 24 in the CSPLib. One considers K sets of the numbers 1 to N. They must be arranged in one sequence so that two consecutive 1 appear one number apart,.... and two consecutive I appear I numbers apart. Naturally if a number is a solution, the number read in the other direction is also a solution; one constraint remove this possibility. The name of a problem is defined by the value of K in French followed by the value of N.

This family is considered p.44 in the paper and p.244 in the book.

4.11. &AUTOREF

A problem of this family was given in the newspaper *Le Monde*. I give a complete description of this problem in the paper p.10, and I mention the results p.44. It generates a sequence of P+2 numbers and the value of the last number must be N. It is a generalization of problem 19 in the CSPLib: when N=0, one finds a magic sequence of length P+1.

4.12.&KAKURO

NC squares appear in NK lines. The squares are numbered from 1 to NC; the goal is to find the value of F{I}which is the content of square I. One must put a number between 1 and 9 in each square so that the sum of the numbers in a line is equal to the value given for this line in a vector H. Moreover, the value of the numbers in the squares belonging to a line must be distinct. The squares belonging to a line are given in a table T.

We can notice that this definition is more extensive that the usual definition of this problem, where more restrictions arise from the geometrical representation with a planar grid. Here, it is possible that a square belongs to more than two lines, and that two lines have more than one square in common.

4.13. &LOGIGRAPH

On a board of H squares by V squares, a sequence of integers is given for each rank and each column. One must blackened some squares of the board so that for each rank and each column the constraint coming from the associated sequence is satisfied.

When the sequence for a line is (x1, x2, ..., xn), that means that one has first zero or any number of white squares, then x1 black squares, then 1 or more white squares, then x2 black squares, then 1 or more white squares, then....., then xn black squares, then zero or more white squares before the end of the line. If a sequence is reduced to (0), that means that there is no black square on this line.

The sequences for the ranks are given in table TH, and for the columns in TV. F{I} indicates the value of square I, 0 for white, and 1 for black. DH and DV are auxiliary functions.

This problem is described in the paper p.35, and an example of a problem with its solution is given p.36. A French site http://mathoeuf.free.fr/lg/public/index.php3 contains many Logigraph problems with their solution.

4.14. &QGROUP

The quasigroup problem is problem 3 in the CSPLib. A quasigroup of order N is a latin square of size N. The goal is to find if a quasigroup of order N with one property C. I have included the following properties:

C=3: (a*b)*(b*a)=a

C=4: (b*a)*(a*b)=a

C=5: ((b*a)*b)*b=a

C=6: (a*b)*b=a*(a*b)

C=7: (b*a)*b=a*(b*a)

One may also add another constraint: the group is idempotent. In this case, M=1 instead of 0.

As usual for this problem, I added the symmetry breaking constraint: a*N>=a-1 for a∈[1:N-1].

This problem is presented in the paper p. 45 to 47. CAIA's results are compared with those obtained by a different method described in a paper by Fujita, Slaney, and Benett which won the award of the best paper at the IJCAI 93.

4.15. &CUBETOTAL

This is a kind of magic cube problem. One must put in the small cubes of as 3x3x3 cube the numbers from 1 to 27 so that the unknown sum of the 9 cubes belonging to 15 planes are the same. These planes are:

6 vertical planes

3 horizontal planes

6 diagonal planes.

There is only one problem in this family. F{I} is the content of square I, and SOM{0} the value of the common sum. The definition of the problem and the results obtained by CAIA are given in the paper, pages 5-8. It is interesting to compare the performances of MALICE to those of any other CSP solver which would only receive the 15 constraints given to MACISTE, and the information that there is a bijection between the small cubes and the integers from 1 to 27.

4.16. &ALLINTERVAL

The all-interval problem is problem 7 in the CSPLib. One must find a permutation of numbers [1:N] such as the absolute value of the differences between two consecutive numbers is a permutation of numbers [1:N-1]. F{I}is the value of element I, and D{J} is the value of the Jth difference. The name of a particular problem is the French name for the value of N.

4.17. &STEINER

The Steiner problem is problem 44 in the CSPLib. The ternary Steiner problem of order N requires to find N(N-1)/6 triples of distinct integers in [1:N] such that any two triples have at most one common element. We have here a generalization of this problem: P is the number of triples and N the number of possible values for the elements, and we remove the constraint between N and P. If N=7, we have P=7 for the strict Steiner problem,; however, we can also consider the case where N=6 and P=7 which is no longer a strict Steiner problem since P is not equal to 6*5/6=5. Another special case is defined by T: if T=0, two triples have at most one common element, but if T=1, they must have exactly one common element.

The names of the problems are N in French followed by the number P followed by 'EQ' when two triples must have exactly one common element.

In the constraints, the value of the function NBVRAI[C1, C2, C3,....., Cn] is the number of conditions Ci which are true. The Ith triple in a solution is: F{I}, G{I}, H{I}.

4.18. &SOCCERTABLE

When a soccer championship is completed, a table indicates for each club the number of wins, draws, losses, goals scored, and goals against. The newspaper *Le Monde* suggested the problem to find from this table the scores of all the matches which have been played if each club played once against every other club.

N is the number of clubs, T is a vector giving the number of wins of each club, X gives the draws, and D the losses. BP indicates the number of goals scored by each club, and BC the number of goals against. The clubs are numbered from 0 to N-1, and the matches from 0 to N*N-1. A match M is played by club I against club J, the formula computing M is: M=I+N*J. For finding a solution, MALICE finds five functions of M, match of club I against club J:

G{M}=1 iff club I won the match, L{M}=1 if the match is a draw, and P{M}=1 iff club I loses the match. MP{M} is the number of goals scored by club I, and MC{M} is the number of goals against club I. Naturally the results are inversed for the value of M corresponding to a match of J against I. All the values corresponding to a match of I against itself have no meaning; they are put to zero.

5. The symmetries

For some problems, CAIA was able to find symmetries. In that case, they were stored when the problem was generated or given; thus, they are found only once. MALICE uses them, adds constraints for breaking them, which do not appear in the formulation. It indicates the number of solutions which it has found in taking these symmetries into account. See for instance &CUBETOTAL or &KAKURO MAL36. For finding some symmetries, CAIA defines a CSP solved by MALICE; this is described p.37-39 in the paper.

For other problems, such as &QGROUP, the constraints breaking the symmetries are among the constraints which were given by the author of the formulation, CAIA is not yet able to find them. MALICE does not know why these constraints were given, and it indicates the number of solutions without taking their symmetrical variants into account.