MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

Artificial Intelligence
Memo No. 188

February 1970

A STABILITY TEST FOR CONFIGURATIONS OF BLOCKS

Manuel Blum, Arnold Griffith, Bernard Neumann

This work is based on notes provided by Manuel Blum, which are paraphrased in section I, and which contain the examples used in the appendix. The main portion of this report was written by Bernard Neumann, who generalized Blum's results to situations involving friction. The program performing the relevant computation, which appears in the appendix, was written by Arnold Griffith, who compiled this memo.

INTRODUCTION

One of the projects in Artificial Intelligence is the block-piling automaton. Given a pile of rectangular blocks on top of each other, the system "looks" at this configuration and mechanically piles up the exact copy, selecting the right blocks from a supply. There is just one mechanical arm which performs the construction of the configuration-copy, so that only one block can be handled at the time. Usually, the order in which these single blocks have to be piled up cannot be uniquely determined in a straight forward manner. The configuration of Fig.1a can clearly be built up in the order 1-2-3-4 as well as 1-2-4-3.

a b

Fig.1

The slight modification in Fig.1b shows what problem can arise. 1-2-3-4 clearly works, but 1-2-4-3 implies the state of construction 1-2-4 which is unstable and hence not realizable. It is therefore necessary that each state of construction is tested for stability before a particular construction strategy is selected.

The existing program handles configurations in which two sides of each (rectangular) block are horizontal. Fig.2 shows a typical pile. Chapter I gives a brief description of how these cases are treated.

The main purpose of this paper is to discuss a generalization of the stability test to arbitrary configurations. Fig.3 shows a typical case. The essential innovation is the occurrence of friction forces. In chapter III the stability criteria are derived, and a program is designed which performs the stability test.

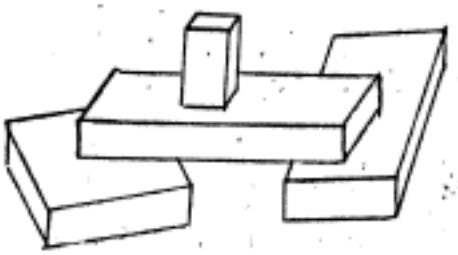


Fig.2

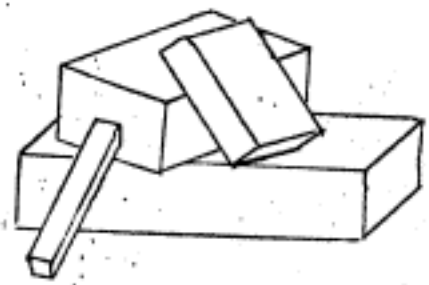


Fig.3

I. CONFIGURATIONS WITHOUT FRICTION

1. Statics

The existing program handles configurations in which two sides of each block are horizontal as shown in Fig.2. This restriction assures that only vertical forces occur, the weights of the rectangular solids and the corresponding reaction forces. By an assumption which reflects the inevitable inaccuracies of the blocks, reaction forces can only occur at

(i) vertices of blocks and

(ii) intersections of line segments

as shown in Fig.4. Fig.5 shows a typical constellation of forces acting on a single block.

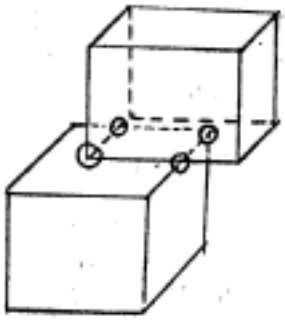


Fig.4

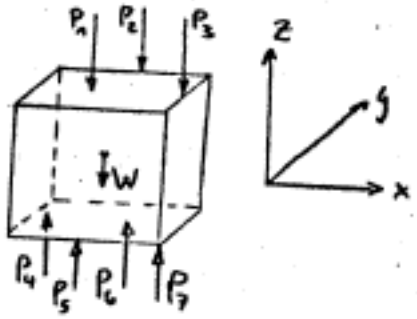


Fig.5

There are three equations which serve to determine unknown forces P_i:

$$\Sigma P_i = 0$$

$$\Sigma M_i^x = 0$$

$$\Sigma M_i^y = 0$$

M_i^x and M_i^y are the moments with respect to the x- and y-axis respectively.

If there are n blocks, we have 3n equations for 3m unknown forces, where $m \geq n$ in general. If $m = n$ the system is called statistically determinate. But usually $m > n$, and the forces cannot be determined uniquely.

2. Stability

A simple rule for stability is the following: given that all forces have been introduced with the directional sense of Fig.5. i.e. pointing towards the block, then the configuration is stable, iff there exists a solution to the 3n equations such that no force is negative. This is quite obvious since a negative force would mean that two blocks try to seperate and the reaction force must keep them together. Since there is no glue or chewing gum in between, this type of force cannot occur and the blocks tumble.

3. Program

The program that executes this stability test works as follows. As preliminary manipulations

(i) the 3n equations for the 3m unknown forces are formulated

(ii) 3(m-n) forces are temporarily treated as constants while the equations are solved for each of the remaining 3n forces

(iii) the solutions are brought into the form shown below where each $b_i \geq 0$.

positive equations

$$x_1 = b_1 + a_{1,3n+1}x_{3n+1} + \cdots + a_{1,m}x_m$$
$$\vdots$$
$$x_k = b_k + a_{k,3n+1}x_{3n+1} + \cdots + a_{k,m}x_m$$

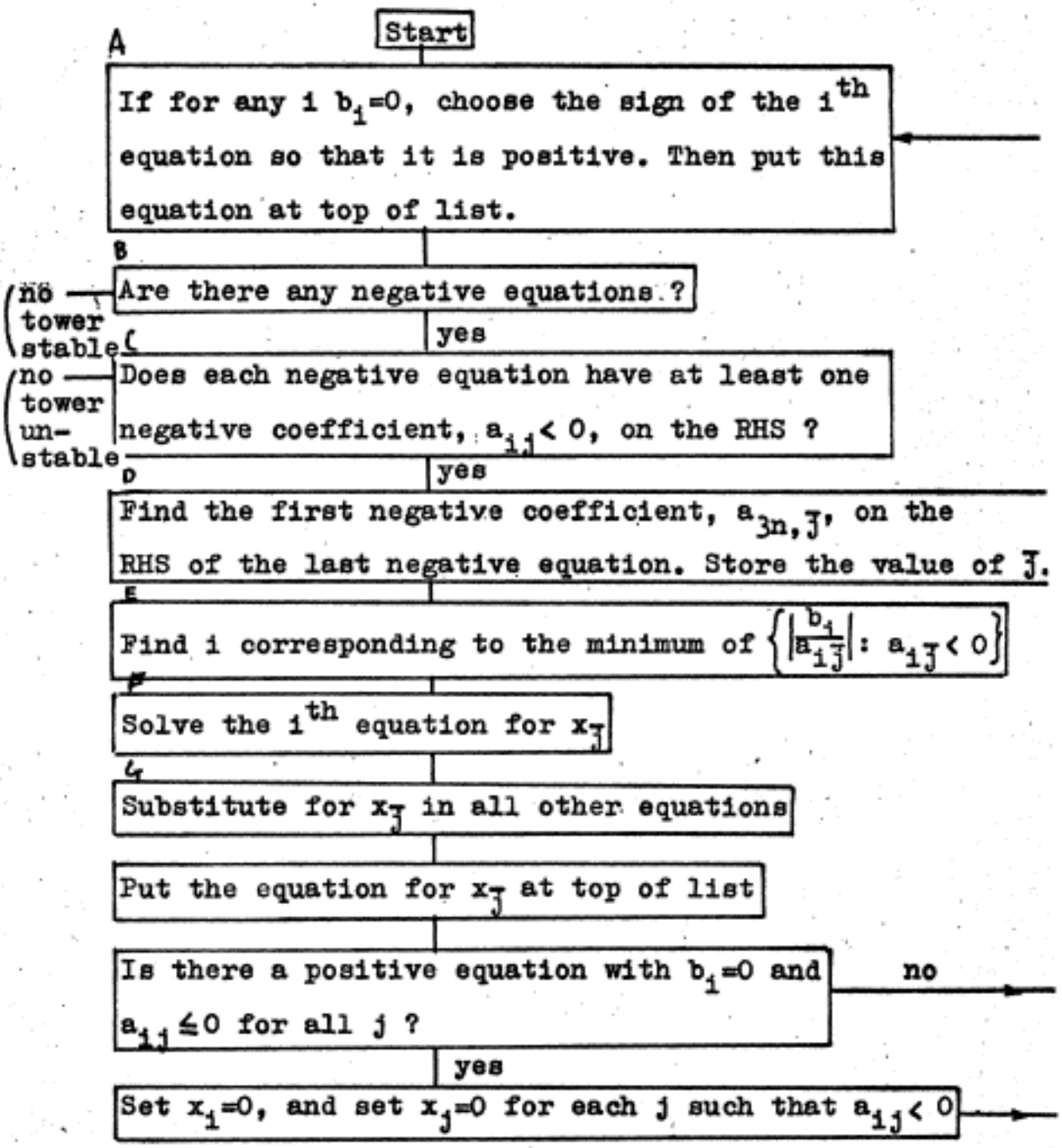
negative equations

$$-x_{k+1} = b_{k+1} + a_{k+1,3n+1}x_{3n+1} + \cdots + a_{k+1,m}x_m$$
$$\vdots$$
$$-x_{3n} = b_{3n} + a_{3n,3n+1}x_{3n+1} + \cdots + a_{3n,m}x_m$$

The actual stability testing program tries to get rid of the "negative equations", because if all equations are positive, the variables on the RHS can be set to zero and the stability requirement is satisfied. This is done by substituting a RHS-variable with negative coefficient for the negative LHS-variable. If one finally ends up, however, with a negative equation which has only positive coefficients on the RHS, this equation can never be satisfied with positive forces and the configuration is unstable.

From a more theoretical point of view the program makes use of the following theorem: Given a block configuration with M unknown forces and described by N equations, then if there exists a stable solution at all, there must be one with M-N forces set to zero.

This is the flow chart of the program:



II. 2-DIMENSIONAL CONFIGURATIONS WITH FRICTION

1. Statics

Before extending the discussion to the general 3-dimensional case with friction we consider an easier situation which has some academic importance, leads to a simple solution and also shows an essential new feature which one encounters in the general case. Fig.6 shows a typical 2-dimensional configuration with friction.

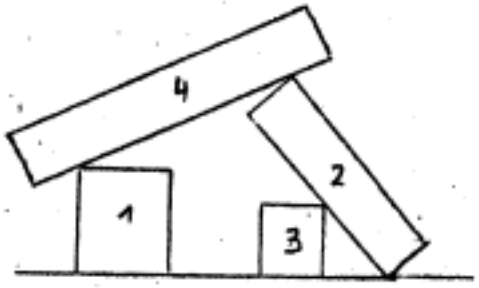


Fig.6

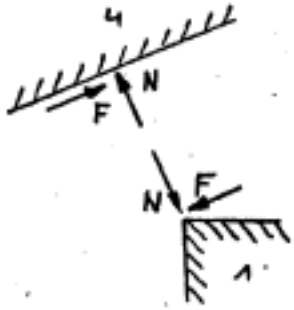


Fig.7

It is intuitively clear that rectangle 4 may or may not slide downwards depending on the friction between 4 and 1, and 4 and 2. The reaction force in the touching point can be decomposed into forces N and F, normal and parallel to the edge of rectangle 4, respectively.(Fig.7). N can only be positive and is quite alike the reaction force encountered in the model without friction. F is the friction force and stands for the resistance of two bodies against sliding on each other, as long as they are still at rest. Clearly this resistance cannot be

unlimited, and depending on the surfaces of the bodies and on the force by which they are pressed on each other, some reaction F might exceed that limit and cannot be "produced" by the system - the blocks start sliding and the configuration is unstable.

Experiments showed that $|F| \leq \mu N$ is a good approximation of this upper limit. μ is a surface parameter and N is the normal force at the touching point. To be consistent with the assumption that forces occur only at isolated spots (and not as line- or area-forces) - an assumption which reflects the geometrical imperfections of the rectangles - we postulate that these isolated spots are the only (point sized) areas where friction occurs. This means in consequence that the geometry around the friction point does not affect the surface parameter. Furthermore we shall assume that the rectangles are made of the same material. Hence μ can be considered constant for the whole system.

There are three equilibrium conditions for each rectangle (Fig.9).

$\Sigma P_x = 0$ forces in the direction of the x-axis

$\Sigma P_z = 0$ forces in the direction of the z-axis

$\Sigma M_y = 0$ moments with respect to the y-axis

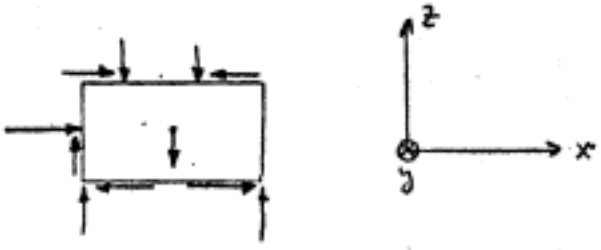


Fig.9

2. Stability

As in chapter I the normal forces must be nonnegative. In addition the friction forces must obey

$$|F_i| \leq \mu N_i \qquad (1)$$

where the F_i may be positive or negative, depending on the direction in which they have been introduced.

3. Program

In this 2-dimensional case with friction it is possible to use the same program as described in chapter I. The trick is to express the condition of Eq.1 in such a way that all forces simply have to be positive for stabilit This can be done by letting

$$F_i + D_i^{(1)} = \mu N_i \qquad (2)$$

$$-F_i + D_i^{(2)} = \mu N_i \qquad (3)$$

Where $D_i^{(1)}$ and $D_i^{(2)}$ both have to be positive for Eq.1 to hold. Hence we simply have to replace each F_i in the equilibrium equations with the help of Eq.2 and add Eq. (derived from Eqs.2 and 3) .

$$D_i^{(1)} + D_i^{(2)} = 2 \mu N_i \qquad (4)$$

The resulting system of equations can be treated exactl as described in chapter I.

III. GENERAL CASE WITH FRICTION

1. Statics

Fig.10 shows a typical configuration of blocks. Let us consider the reaction forces between block 1 and 2. Following the convention of chapter I we locate the only reaction forces at the encirceled spots.(Fig.11).

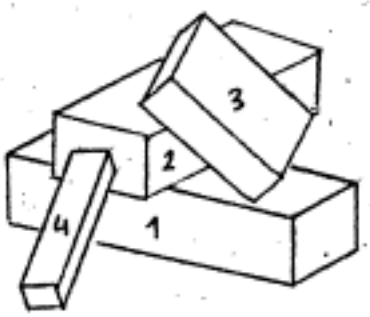


Fig.10

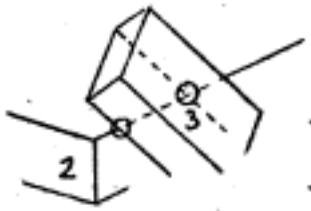


Fig.11

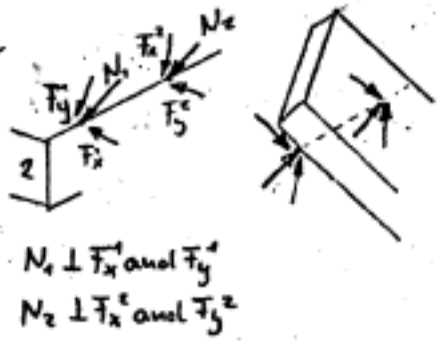


Fig.12

At each of the spots we have a normal force and a friction force. Unfortunately, because of the additional dimension, the direction of each friction force is unknown 2-dimensionally, so that we must introduce two orthogonal components, F_x and F_y. Fig.12 shows the reaction forces. The resulting friction force is then

$$|F| = \sqrt{F_x^2 + F_y^2} \qquad (5)$$

and is limited again by

$$|F| \leq \mu N \qquad (6)$$

As before μ is assumed to be constant for the whole configuration.

The statics of a single block are described by the full set of six equations.

$$\sum P_x = 0 \qquad \sum M_x = 0$$
$$\sum P_y = 0 \qquad \sum M_y = 0$$
$$\sum P_z = 0 \qquad \sum M_z = 0$$

Note that Eq.6 does not serve to determine unknown forces but is a control inequality which tells when the described system is unstable, i.e. unrealizable in the given form.

2. Stability

The stability criteria are as before. All normal forces must be positive, and all friction forces must obey Eqs.5 and 6. It is evident that a nonlinear relation enters the calculations. This constitutes the essential difference between the general case with friction on the one hand and the 2-dimensional case with friction as well as the 3-dimensional case without friction on the other hand.

3. Program

There are several ways of executing the stability test. One strategy is to use the linear equations and search for a solution which is stable in the normal forces. If there exists such a solution one needs only plug the values of the friction forces into the inequalities of

the type $$\sqrt{F_x^2 + F_y^2} \leq \mu N \qquad (7)$$

If they are satisfied, the configuration is stable. If they are not, there may be three reasons.

(i) Friction forces and normal forces have been discarded as statically redundant in an inconsistent manner. Fig.13 shows such a case. By some additional complexity, however, the program can be modified to avoid this case.

(ii) Forces have been discarded consistently, but still another choice of redundant forces may lead to the result "system stable". Fig.14 shows this pathological case. The normal force at either a or b has to be discarded. If b is assumed to carry, block 3 might not slide on 2. If however the normal force at b is discarded, block 3 gets additional load at a and the inequality will indicate "system unstable". It seems impossible to avoid this pitfall and still have an efficient and fast stability test.ir the basic strategy of the program is kept.

(iii) The configuration is in fact unstable - the desired result.

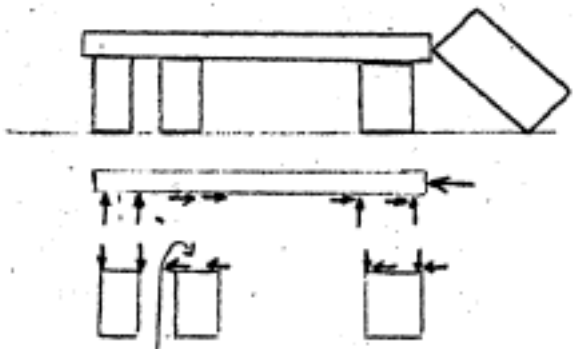


Fig.13

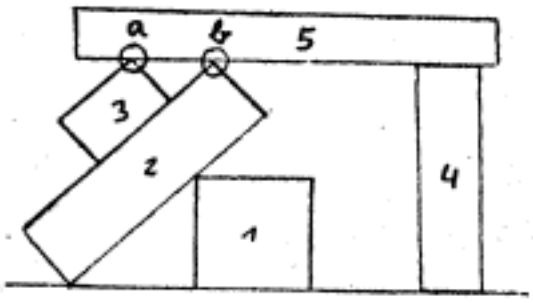


Fig.14

The complexity of other ways of executing the stability test is mainly due to the nonlinearity of Eq.7. I believe, however, that this effort - on the part of the programmer - and this additional computation time - on the part of the program - is not really justified, because the upper limit of friction forces given by Eq.6 is a rather crude approximation of a physically complex relationship. The surface parameter μ, moreover, is not really well-defined but depends on each microscopic scratch which may or may not be at a particular spot. In short: If a reasonable approximation of the nonlinearity of Eq.7 can be found, it will not degrade the overall performance decisively because it is not the first approximation.

Fig.15 shows the region for which $F_x^2 + F_y^2 \leq (\mu N)^2$.

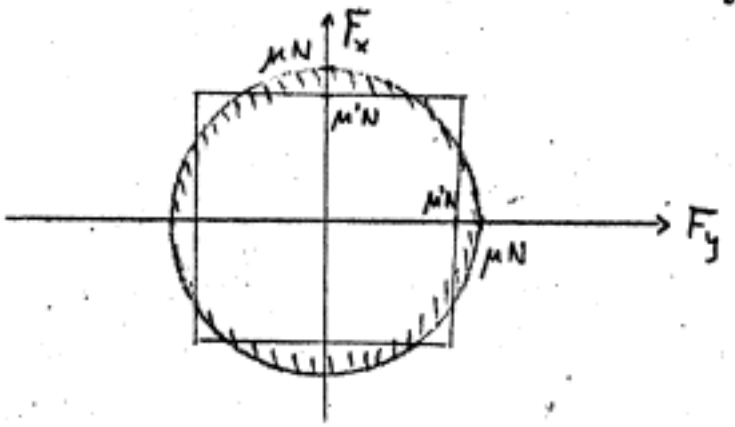


Fig.15

If one approximates the circle by a square, the conditions read $|F_x| \leq \mu' N$

and $|F_y| \leq \mu' N$

Now we are back to a linear restriction and can write as in chapter II

$$F_x + D_x^{(1)} = \mu' N \qquad -F_x + D_x^{(2)} = \mu' N$$
$$F_y + D_y^{(1)} = \mu' N \qquad -F_y + D_y^{(2)} = \mu' N$$

where all D's must be positive for the restrictions to be satisfied.

If we replace each F_i by $N_i - D_i^{(1)}$ and add the equation $D_i^{(1)} + D_i^{(2)} = 2\ N_i$, The original program as described in chapter I will perform the stability test, and the extension to the general case with friction is carried out quite painlessly.

APPENDIX

I. FUNCTIONS FOR SOLVING STABILITY EQUATIONS

(SUP1 X) Takes as input a set of equations in the format illustrated in the examples at the end of section III. SUP1 reads into core the print names of the variables appearing in the equations, as well as their coefficients. The coefficients are stored in a matrix whose row elements are the coefficients in order of the corresponding equation expressed in the form:

$$a_{i,1}x_1 + a_{i,2}x_2 + \ldots + a_{i,n}x_n + b_i = 0.$$

(SETUP) Brings the equations implicitly into the form shown in iii) on page 5, by applying appropriate row operations to the coefficient matrix. A row corresponding to an equation solved for a variable x_n in terms of variables $x_1, \ldots x_m$ appears as follows:

$$c_1, c_2, \ldots c_m, 0, \ldots 0, \pm 1, 0, \ldots 0, b,$$

where the ± 1 appears in the nth column, and the value of b is positive. A +1 in the nth comumn indicates a "negative" equation.

(LOOP) Applies the algorithm flowcharted on page 6. The letters on the boxes in the figure correspond to tags in the program as listed in section III. The program either outputs the atom UNSTABLE, the atom STABLE, or the atom CONTINUE. In the latter case, another iteration of the LOOP program is indicated.

(OUTPUT) Outputs a formatted version of the current state of the coefficient matrix.

II. EXAMPLES

Equations corresponding to these examples appear at the end of section III.

Each block four units square in the following diagrams (Figures 16 - 19) are considered to have one unit of mass. The points at which the forces act are given letters A, B, etc., which are used as the corresponding force variables in the equations.

EX1

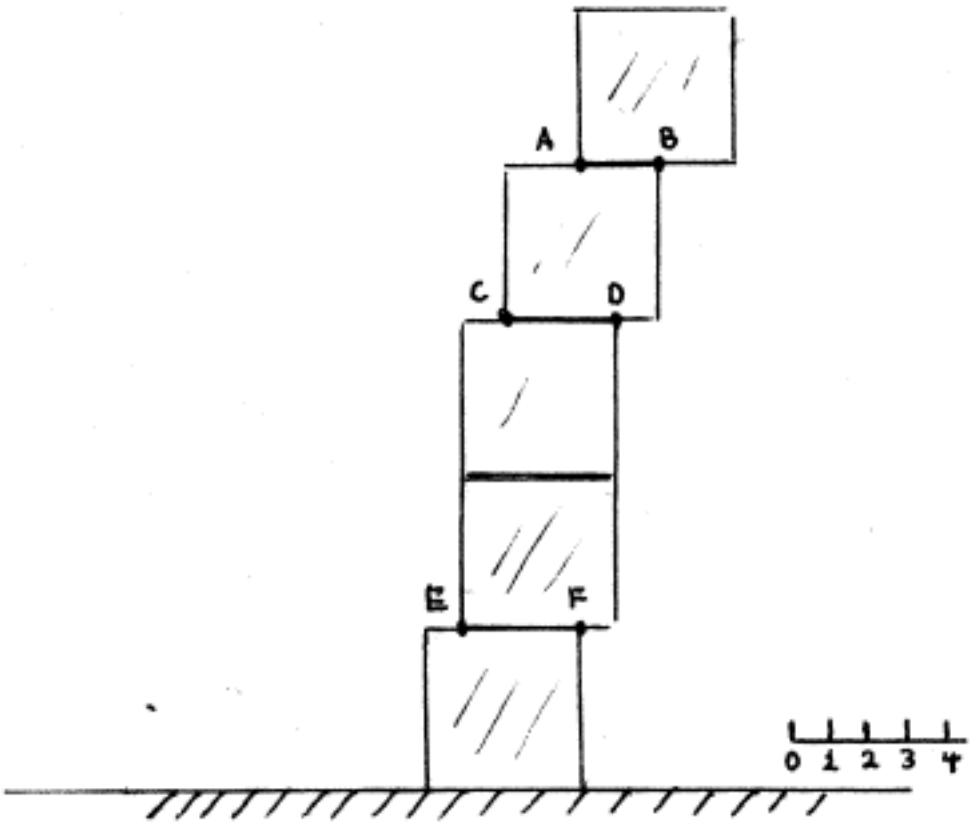


Figure 16

The force (F) and moment (M) equations for the situation in figure 16 are:

F: A + B = 1

M: 2A = 0

F: C + D = 2

M: 6 - 3D = 0

F: F + E = 4

M: 12 - 3F = 0.

After executing the program SUP1, the matrix of coefficients is:

```
 NIL    A      B      C      D      F      E      NIL
 0       1.00   1.00    .00    .00    .00    .00  -1.00
 0       2.00    .00    .00    .00    .00    .00    .00
 0        .00    .00   1.00   1.00    .00    .00  -2.00
 0        .00    .00    .00  -3.00    .00    .00   6.00
 0        .00    .00    .00    .00   1.00   1.00  -4.00
 0        .00    .00    .00    .00  -3.00    .00  12.00
NIL

NIL
```

After executing SETUP, the coefficients are:

```
 NIL    A      B      C      D      F      E      NIL
 2        .00  -1.00    .00    .00    .00    .00   1.00
 1       1.00    .00    .00    .00    .00    .00    .00
 3        .00    .00   1.00    .00    .00    .00    .00
 4        .00    .00    .00  -1.00    .00    .00   2.00
 6        .00    .00    .00    .00    .00   1.00    .00
 5        .00    .00    .00    .00  -1.00    .00   4.00
6

STABLE
```

The numbers in the leftmost vertical column are the serial numbers of the variable which are solved for. For example, the first equation above is solved for the second variable, B.

Upon execution of LOOP, the atom STABLE is output, because all equations have been reduced to positive ones:

NIL	A	B	C	D	F	E	NIL
2	.00	-1.00	.00	.00	.00	.00	1.00
1	-1.00	.00	.00	.00	.00	.00	.00
3	.00	.00	-1.00	.00	.00	.00	.00
4	.00	.00	.00	-1.00	.00	.00	2.00
6	.00	.00	.00	.00	.00	-1.00	.00
5	.00	.00	.00	.00	-1.00	.00	4.00

EX2

As before, the square blocks are considered to have unit mass:

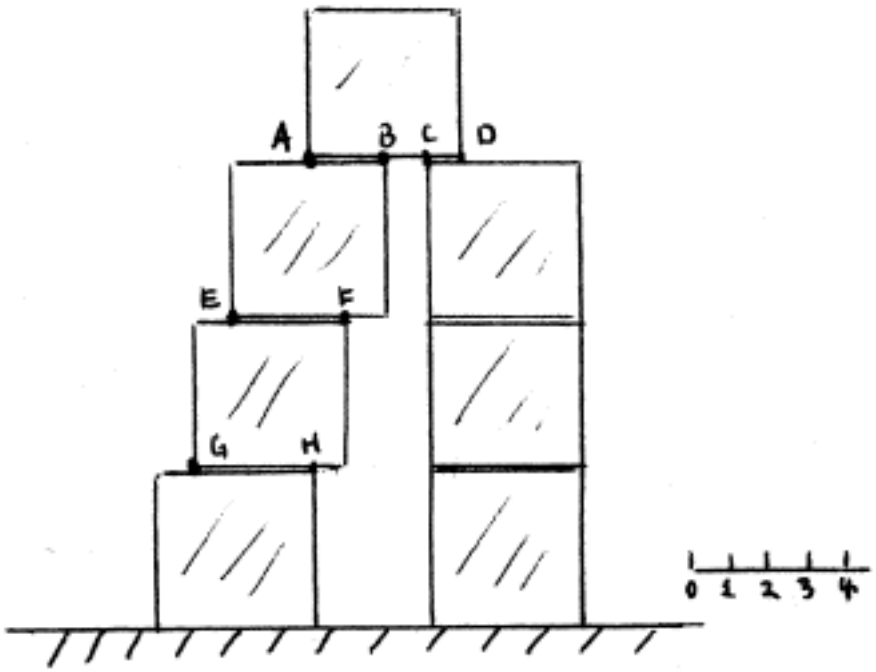


Figure 17.

The equations for this situation are:

F: $A + B + C + D = 1$

M: $2 - 2B - 3C - 4D = 0$

F: $E + F = 1 + A + B$

M: $3F - 2 - 2A - 4B = 0$

F: $G + H = 1 + E + F$

M: $E + 2 + 4F - 3H = 0$

The results of applying SUPI, SETUP, and LOOP (three times) are:

NIL	A	B	C	D	E	F	G	H	NIL
0	1.00	1.00	1.00	1.00	.00	.00	.00	.00	-1.00
0	.00	-2.00	-3.00	-4.00	.00	.00	.00	.00	2.00
0	-1.00	-1.00	.00	.00	1.00	1.00	.00	.00	-1.00
0	-2.00	-4.00	.00	.00	.00	3.00	.00	.00	-2.00
0	.00	.00	.00	.00	-1.00	-1.00	1.00	1.00	-1.00
0	.00	.00	.00	.00	1.00	4.00	.00	-3.00	2.00
NIL									

NIL	A	B	C	D	E	F	G	H	NIL
3	-4.00	-2.00	-1.00	.00	.00	.00	.00	.00	2.00
4	-3.00	-1.00	.00	1.00	.00	.00	.00	.00	1.00
5	.33	-.33	.00	.00	-1.00	.00	.00	.00	.33
6	.66	1.33	.00	.00	.00	-1.00	.00	.00	.66
7	.00	-.66	.00	.00	.00	.00	-1.00	.00	.33
8	.99	1.66	.00	.00	.00	.00	.00	-1.00	1.66
8									

NIL	A	B	C	D	E	F	G	H	NIL
2	.00	-1.00	.00	.00	.00	.00	-1.50	.00	.50
4	-3.00	.00	.00	1.00	.00	.00	1.50	.00	.49
5	.33	.00	.00	.00	-1.00	.00	.50	.00	.16
6	.66	.00	.00	.00	.00	-1.00	-2.00	.00	1.33
3	-4.00	.00	-1.00	.00	.00	.00	3.00	.00	.99
8	.99	.00	.00	.00	.00	.00	-2.50	-1.00	2.50
8									

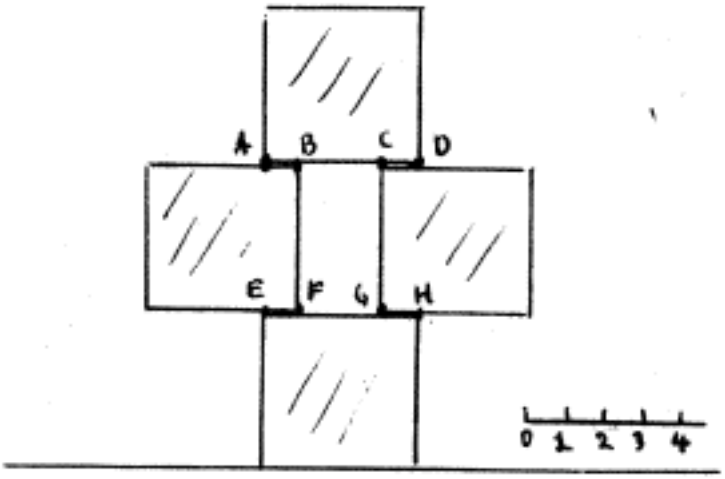
NIL	A	B	C	D	E	F	G	H	NIL
1	-1.00	.00	.00	.33	.00	.00	.50	.00	.16
2	.00	-1.00	.00	.00	.00	.00	-1.50	.00	.50
5	.00	.00	.00	.11	-1.00	.00	.66	.00	.22
6	.00	.00	.00	.22	.00	-1.00	-1.66	.00	1.44
3	.00	.00	-1.00	-1.33	.00	.00	1.00	.00	.33
8	.00	.00	.00	.33	.00	.00	-1.99	-1.00	2.66
8									

NIL	A	B	C	D	E	F	G	H	NIL
1	-1.00	.00	.00	.33	.00	.00	.50	.00	.16
2	.00	-1.00	.00	.00	.00	.00	-1.50	.00	.50
5	.00	.00	.00	.11	-1.00	.00	.66	.00	.22
6	.00	.00	.00	.22	.00	-1.00	-1.66	.00	1.44
3	.00	.00	-1.00	-1.33	.00	.00	1.00	.00	.33
8	.00	.00	.00	.33	.00	.00	-1.99	-1.00	2.66
8									

At the end of the third iteration, the LOOP program output the atom STABLE.

EX3

The situation is as follows:



The force and moment equations:

F: $A + B + C + D = 1$

M: $2A + B - C - 2D = 0$

F: $A + B + 1 = E + F$

M: $A - E + 2B - 2F = 0$

F: $C + D + 1 = G + H$

M: $2C - 2G + D - H = 0$

The states of the coefficient matrix after SUP1, SETUP, and 2 iterations of LOOP:

NIL	A	B	C	D	E	F	G	H	NIL
0	1.00	1.00	1.00	1.00	.00	.00	.00	.00	-1.00
0	2.00	1.00	-1.00	-2.00	.00	.00	.00	.00	.00
0	1.00	1.00	.00	.00	-1.00	-1.00	.00	.00	1.00
0	1.00	2.00	.00	.00	-1.00	-2.00	.00	.00	.00
0	.00	.00	1.00	1.00	.00	.00	-1.00	-1.00	1.00
0	.00	.00	2.00	1.00	.00	.00	-2.00	-1.00	.00
NIL									

NIL	A	B	C	D	E	F	G	H	NIL
3	-4.00	-3.00	-1.00	.00	.00	.00	.00	.00	2.00
4	-3.00	-2.00	.00	1.00	.00	.00	.00	.00	1.00
5	1.00	.00	.00	.00	-1.00	.00	.00	.00	2.00
6	.00	-1.00	.00	.00	.00	1.00	.00	.00	1.00
7	-4.00	-3.00	.00	.00	.00	.00	-1.00	.00	1.00
8	3.00	2.00	.00	.00	.00	.00	.00	-1.00	1.00

8

NIL	A	B	C	D	E	F	G	H	NIL
2	-1.33	-1.00	.00	.00	.00	.00	-.33	.00	.33
4	-.33	.00	.00	1.00	.00	.00	.66	.00	.33
5	1.00	.00	.00	.00	-1.00	.00	.00	.00	2.00
6	1.33	.00	.00	.00	.00	1.00	.33	.00	.66
3	.00	.00	-1.00	.00	.00	.00	1.00	.00	1.00
8	.33	.00	.00	.00	.00	.00	-.66	-1.00	1.66

8

UNSTABLE

NIL	A	B	C	D	E	F	G	H	NIL
2	-1.33	-1.00	.00	.00	.00	.00	-.33	.00	.33
4	-.33	.00	.00	1.00	.00	.00	.66	.00	.33
5	1.00	.00	.00	.00	-1.00	.00	.00	.00	2.00
6	1.33	.00	.00	.00	.00	1.00	.33	.00	.66
3	.00	.00	-1.00	.00	.00	.00	1.00	.00	1.00
8	.33	.00	.00	.00	.00	.00	-.66	-1.00	1.66

8

After the second iteration, the atom UNSTABLE was output by LOOP.

EX9

The situation is shown in figure 19. The square blocks have unit mass, and the others have mass proportional to their areas.

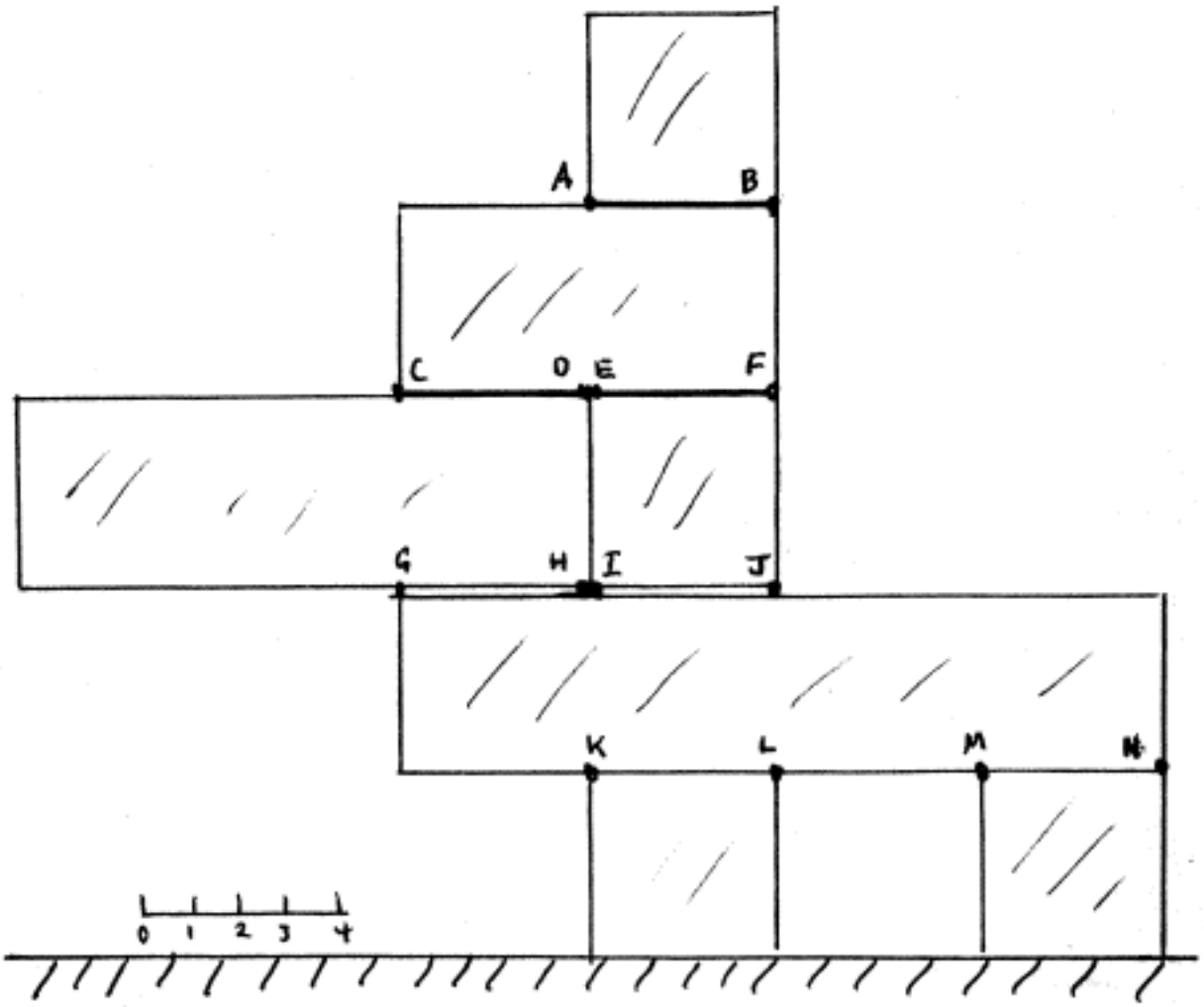


Figure 19.

The force and moment equations:

F: A + B = 1

M: ½ − B = 0

F: A + B + 2 = C + D + E + F

M: A + 2 + 2B = D + E + 2F

F: C + D + 3 = G + H

M: ½C + 1½D = ½G + 1½H

F: E + F + 1 = I + J

M: ½ + F = J

F: G + H + I + J + 4 = K + L + M + N

M: 2G + H + I − K + M + 2N = D

The solution to this stability problem, by means of the programs given, is left as an exercise.

III. LISTINGS

```
(DEFPROP SUP1(LAMBDA(X)(PROG(I J VLIST EQS VAR)
 (SETQ I 0)
 (SETQ X(MAPCAR(FUNCTION(LAMBDA(Y)(CONS(SETQ I(ADD1 I))Y)))X))
 (SETQ EQS I)
 (SETQ I 0)
 (SETQ VLIST NIL)
 (MAPC(FUNCTION(LAMBDA(Y)
  (MAPC(FUNCTION(LAMBDA(Z)
   (AND(NOT(ASSOC(CAR Z)VLIST))
        (SETQ VLIST(CONS(CONS(CAR Z)(SETQ I(ADD1 I)))VLIST)))))
       (CDDR Y))))
      X)
 (SETQ VAR I)
 (DEPOSIT(GET(QUOTE EQS)(QUOTE SYM))EQS)
 (DEPOSIT(GET(QUOTE VAR)(QUOTE SYM))VAR)
 (COND((GREATERP(SETQ I(*DIF
         (PLUS BPORG(TIMES(ADD1 EQS)(PLUS 2 VAR)))
         BPEND))0)
       (RETURN(CONS I(QUOTE(MORE BP WORDS NEEDED))))))
 (COND((GREATERP EQS 50.)(RETURN(QUOTE(TOO MANY EQIATIONS)))))
 (SETQ I 0)
RT(DEPOSIT(PLUS(GET(QUOTE IND2)(QUOTE SYM))I)
    (BOOLE 7 2000000(PLUS BPORG(TIMES I(ADD1 EQS)))))
  (DEPOSIT(PLUS(GET(QUOTE IND3)(QUOTE SYM))I)
    (BOOLE 7 3000000(PLUS BPORG(TIMES I(ADD1 EQS)))))
 (COND((GREATERP(SETQ I(AUD1 I))(ADD1 VAR))(GO T1)))
 (GO RT)
T1(DEPOSIT (GET(QUOTE INDC2)(QUOTE SYM))(BOOLE 7 2000000
     (PLUS BPORG(TIMES(ADD1 EQS)(ADD1 VAR)))))
  (DEPOSIT (GET(QUOTE INDP2)(QUOTE SYM))(BOOLE 7 2000000 BPORS))
(CLEAR)
 (MAPC(FUNCTION(LAMBDA(Y)
  (MAPC(FUNCTION(LAMBDA(Z)
   (INSERT(CDR(ASSOC(CAR Z)VLIST))(CAR Y)(CDR Z)) ))
       (CDDR Y))))
      X)
 (MAPC(FUNCTION(LAMBDA(Y)(INSERT(ADD1 VAR)(CAR Y)(CADR Y) )))
      X)
 (MAPC(FUNCTION(LAMBDA(X)(INSERT(CDR X)0(CAR X))))VLIST)
 (OUTPUT)
 (RETURN NIL) ))EXPR)

(MAPC(FUNCTION(LAMBDA(X Y)(PUTPROP X(PLUS Y BPORG)(QUOTE SYM))))
 (QUOTE (EQS VAR INDP2 INDC2 IND2 IND3))
 (QUOTE(0 1 2 3 4 54. )))

(SETQ BPORG (PLUS BPORG 104.))
```

```
(LAP SETUP SUBR)
S       (MOVE 2 EQS)
S1      (MOVE 1 VAR)
S2      (SKIPN@ 0 IND2 1)
        (SOJG 1 S2)
        (MOVEM@ 1 INDP2)
        (CALL 2(QUOTE ROWSUB))
        (SOJG 2 S1)
        (MOVE 2 EQS)
S3       (MOVE@ 3 INDC2)
        (JUMPGE 3 S5)
        (MOVE 1 VAR)
        (ADDI 1 1)
S4      (MOVE@ 3 IND2 1)
        (MOVNM@ 3 IND2 1)
        (SOJG 1 S4)
S5      (SOJG 2 S3)
        (POPJ P)
  NIL

(LAP LOOP SUBR)
A       (MOVE 2 EQS)
A1      (MOVE@ 3 INDC2)
        (JUMPN 3 A3)
        (MOVE@ 1 INDP2)
        (MOVE@ 3 IND2 1)
        (JUMPL 3 A3)
        (MOVE 1 VAR)
        (ADDI 1 1)
A2      (MOVE@ 3 IND2 1)
        (MOVNM@ 3 IND2 1)
        (SOJG 1 A2)
A3      (SOJG 2 A1)
B       (MOVE 2 EQS)
B1      (MOVE@ 1 INDP2)
        (MOVE@ 3 IND2 1)
        (JUMPGE 3 B2)
        (SOJG 2 B1)
        (MOVEI 1 (QUOTE STABLE))
        (POPJ P)
B2      (MOVEM 2 NEGEQ)
C       (MOVE 2 EQS)
C1      (MOVE 1 VAR)
C2      (MOVE@ 3 IND2 1)
        (JUMPL 3 C3)
        (SOJG 1 C2)
        (MOVEI 1(QUOTE UNSTABLE))
        (POPJ P)
C3      (SOJG 2 C1)
```

```
D       (MOVE 2 NEGEQ)
        (MOVE 1 VAR)
D1      (SKIPL@ 0 IND2 1)
        (SOJG 1 D1)
E       (MOVE 4 BIG)
        (MOVE 2 EQS)
E1      (MOVN@ 3 IND2 1)
        (JUMPLE 3 E2)
        (MOVE@ 6 INDC2)
        (FDV 6 3)
        (CAML 6 4)
        (JRST 0 E2)
        (MOVE 5 2)
        (MOVE 4 6)
E2      (SOJG 2 E1)
        (MOVE 2 5)
        (MOVEM@ 1 INDP2)
        (CALL 2(QUOTE ROWSUB))
F       (MOVE 1 VAR)
        (ADDI 1 1)
        (MOVEI  3 1)
F1      (MOVE@ 4 IND2 1)
        (EXCH@ 4 IND3 1)
        (EXCH@ 4 IND2 1)
        (SOJGE 1 F1)
G       (MOVE 2 EQS)
G1      (MOVE 1 VAR)
        (ADDI 1 1)
G2      (MOVE@ 3 IND2 1)
        (JUMPG 3 G6)
        (SOJG 1 G2)
        (MOVE 1 VAR)
G3      (MOVE@ 3 IND2 1)
        (JUMPE 3 G5)
        (MOVE 3 EQS)
G4      (SETZM@ 0 IND3 1)
        (SOJG 3 G4)
G5      (SOJG 1 G3)
        (MOVE@ 1 INDP2)
        (MOVE 3 ONE)
        (MOVNM@ 3 IND2 1)
G6      (SOJG 2 G1)
        (MOVE 2 EQS)
S3       (MOVE@ 3 INDC2)
        (JUMPGE 3 S5)
        (MOVE 1 VAR)
        (ADDI 1 1)
S4      (MOVE@ 3 IND2 1)
        (MOVNM@ 3 IND2 1)
        (SOJG 1 S4)
S5      (SOJG 2 S3)
        (MOVEI 1(QUOTE CONTINUE))
        (POPJ P)
BIG     (377777777777)
ONE     (201400000000)
NEGEQ   (0)
 NIL
```

```
      (OPS
 SETZM@ 402020000000
 MOVM@  214020000000)

(LAP ROWSUB SUBR)
       (MOVEM 1 R4)
       (MOVE 3 R5)
       (FDV@ 3 IND2 1)
       (MOVE 1 VAR)
       (ADDI 1 1)
R1     (FMPM@ 3 IND2 1)
       (SOJG 1 R1)
        (MOVE 1 R4)
        (MOVE 3 R5)
        (MOVEM@ 3 IND2 1)
       (MOVE 3 EQS)
R2     (CAMN 3 2)
       (JRST 0 R6)
       (MOVE 1 R4)
       (MOVE@ 4 IND3 1)
       (JUMPE 4 R6)
       (MOVE 1 VAR)
       (ADDI 1 1)
R3     (MOVE@ 5 IND2 1)
       (JUMPE 5 R7)
       (FMP 5 4)
       (FSB@ 5 IND3 1)
       (MOVNM@ 5 IND3 1)
R7     (SOJG 1 R3)
R6     (SOJG 3 R2)
      (MOVE 1 VAR)
      (ADDI 1 1)
R8    (MOVE 3 EQS)
R9    (MOVM@ 4 IND3 1)
      (CAMG 4 R10)
      (SETZM@ 0 IND3 1)
      (SOJG 3 R9)
      (SOJG 1 R8)
      (POPJ P)
R10   (1604000000000)
R4     (0)
R5     (201400000000) NIL
```

```
(LAP OUTPUT SUBR)
        (MOVEI 2 0)
T1      (MOVEI 1 0)
T2      (MOVE* 3 IND2 1)
        (MOVEM 3 T3)
        (MOVEM 2 T5)
        (MOVEM  1 T4)
        (MOVE* 1 IND2 1)
        (MOVEI 2(QUOTE FLONJM))
        (CALL 2(QUOTE MAKNUM))
        (MOVE 4 1)
        (MOVE 3 T3)
        (MOVE 1 T4)
        (MOVE 2 T5)
        (CALL 4(QUOTE OUTPUT*))
        (MOVE 1 T4)
        (MOVE 2 T5)
        (CAMG 1 VAR)
        (AOJA 1 T2)
        (ADDI 2 1)
        (CAMLE 2 EOS)
        (POPJ P)
        (JRST 0 T1)
T3      (0)
T4      (0)
T5      (0) NIL
```

```
(LAP CLEAR SUBR)
       (MOVE 2 EQS)
C1     (MOVE 1 VAR)
       (ADDI 1 1)
C2     (SETZM@ 0 IND2 1)
       (SOJGE 1 C2)
       (SOJGE 2 C1)
       (POPJ P) NIL

(LAP INSERT SUBR)
       (SUBI 1 1)
       (SUBI 2 1)
       (JUMPE 2 I1)
       (PUSH P 1)
       (PUSH P 2)
       (MOVE 1 3)
       (PUSHJ P NUMVAL)
       (MOVE 3 1)
       (POP P 2)
       (POP P 1)
I1     (MOVEM@ 3 IND2 1)
       (POPJ P)
   NIL
```

```
(DEFPROP OUTPUT*(LAMBDA(X Y Z W)(PROG(I BASE *NOPOINT LINEL)
 (SETQ BASE 10.)
 (SETQ *NOPOINT T)
 (SETQ LINEL 120.)
 (COND((AND X Y)(SPT W)(RETURN NIL)))
 (SETQ I (COND(Y(MAKNUM Z(QUOTE FIXNUM)))
               (X Z) ))
 (AND(NOT X)(NOT Y)(TYO 14))
 (AND(NOT X)(TERPRI))
 (PRINC(QUOTE / ))
 (PRINC I)
 (SETQ I(*DIF 5(FLATSIZE I)))
RT(COND((MINUSP(SETQ I(SUB1 I)))(RETURN NIL)))
 (PRINC(QUOTE / ))
 (GO RT) ))EXPR)

(DEFPROP SPT (LAMBDA(X)(PROG(I BASE *NOPOINT J)
 (SETQ *NOPOINT T)
 (SETQ BASE 10.)
 (COND((MINUSP X)(SETQ X(TIMES X -1))(SETQ J(QUOTE /-)))
       (T(SETQ J(QUOTE / ))))
 (SETQ I(EXPLODE(SETQ X(FIX(TIMES 100. X)))))
 (COND
  ((GREATERP X 9999.)(PRINC(QUOTE / / BIG/ )))
  ((GREATERP X 999.)
   (PRINC J)
   (PRINC(CAR I))
   (PRINC(CADR I))
   (PRINC(QUOTE /.))
   (PRINC(CADDR I))
   (PRINC(CADDDR I)) )
  ((GREATERP X 99.)
   (PRINC (QUOTE / ))
   (PRINC J)
   (PRINC(CAR I))
   (PRINC(QUOTE /.))
   (PRINC(CADR I))
   (PRINC(CADDR I)) )
 ((GREATERP X 9.)
   (PRINC(QUOTE / / ))
   (PRINC J)
   (PRINC(QUOTE /.))
   (PRINC(CAR I))
   (PRINC(CADR I)) )
 (T
   (PRINC(QUOTE / / ))
   (PRINC J)
   (PRINC(QUOTE /.0))
   (PRINC(CAR I)) ) )
 (RETURN NIL) ))EXPR)
```

```
(SETQ EX1(QUOTE(
 (-1.0 (A .  1.0)(B .  1.0))
 ( 0.0 (A .  2.0))
 (-2.0 (C .  1.0)(D .  1.0))
 ( 6.0 (D . -3.0))
 (-4.0 (F .  1.0)(E .  1.0))
 (12.0 (F . -3.0)) )))

(SETQ EX2(QUOTE(
 (-1.0 (A .  1.0)(B .  1.0)(C .  1.0)(D .  1.0))
 ( 2.0 (B . -2.0)(C . -3.0)(D . -4.0))
 (-1.0 (A . -1.0)(B . -1.0)(E .  1.0)(F .  1.0))
 (-2.0 (A . -2.0)(B . -4.0)(F .  3.0))
 (-1.0 (E . -1.0)(F . -1.0)(G .  1.0)(H .  1.0))
 ( 2.0 (E .  1.0)(F .  4.0)(H . -3.0)) )))

(SETQ EX3(QUOTE(
 (-1.0 (A .  1.0)(B .  1.0)(C .  1.0)(D .  1.0))
 ( 0.0 (A .  2.0)(B .  1.0)(C . -1.0)(D . -2.0))
 ( 1.0 (A .  1.0)(B .  1.0)(E . -1.0)(F . -1.0))
 ( 0.0 (A .  1.0)(E . -1.0)(B .  2.0)(F . -2.0))
 ( 1.0 (C .  1.0)(D .  1.0)(G . -1.0)(H . -1.0))
 ( 0.0 (C .  2.0)(G . -2.0)(D .  1.0)(H . -1.0)) )))
(SETQ EX9(QUOTE(
 (-1.0 (A .  1.0)(B .  1.0))
 ( 0.5 (B . -1.0))
 ( 2.0(A .  1.0)(B .  1.0)(C . -1.0)(D . -1.0)(E . -1.0)(F . -1.0))
 ( 2.0(A .  1.0)(B .  2.0)(D . -1.0)(E . -1.0)(F . -2.0))
 ( 3.0(C .  1.0)(D .  1.0)(G . -1.0)(H . -1.0))
 ( 0.0(C .  0.5)(D .  1.5)(G . -0.5)(H . -1.5))
 ( 1.0(E .  1.0)(F .  1.0)(I . -1.0)(J . -1.0))
 ( 0.5(F .  1.0)(J . -1.0))
 ( 4.0(G .  1.0)(H .  1.0)(I .  1.0)
      (J .  1.0)(K . -1.0)(L . -1.0)(M . -1.0)(N . -1.0))
 ( 0.0(G .  2.0)(H .  1.0)(I .  1.0)(K . -1.0)(M .  1.0)(N .  2.0)) )))
```