CNEA-NT 4/81
CAB-NT 2/81

REPUBLICA ARGENTINA

COMISION NACIONAL DE ENERGIA ATOMICA

Dependiente de la Presidencia de la Nación

CENTRO ATOMICO BARILOCHE

AUTOMATIC GENERATION OF TWO-DIMENSIONAL ENCLOSURES BETWEEN BODIES IN CONTACT. APPLICATION TO HEAT RADIATION

Computer program "SPADE"

N.H. Callwood and S. Pissanetzky

Buenos Aires

1981

ABSTRACT

Computer programs which use the finite element method often require large volumes of input data in order to define the system under consideration; for example: coordinates of the nodes, connectivity, and physical properties of the elements. Pre-processors for the automatic generation of these data are becoming popular, and this report describes such a pre-processor for a special case: given a set of solid bodies with a two-dimensional geometry, the free zones between these bodies are found, and expressed in terms suitable for input to a finite element program. Practical applications include the description of the heat radiation enclosures between a set of bodies at various temperatures, for the solution of the heat transfer problem /1/.

Logical variables are used throughout the program, thus ensuring that EXACT decisions are taken when it is necessary to determine whether, for example, three points are collinear, or two bodies are in contact. This report describes the method used, and the preparation of input data. Simple examples are included.

RESUMEN

Los programas de computadora que usen el método de Elementos Finitos generalmente necesitan una gran cantidad de datos, sólo para definir el sistema en estudio; por ejemplo: coordenadas nodales, conectividad y propiedades físicas de los materiales. La generación automática de estos datos se está haciendo cada vez más popular, y este informe describe un programa que genera automáticamente los datos necesarios para el siguiente caso: dado un conjunto de cuerpos sólidos con una geometría bidimensional, el programa encuentra las zonas libres entre los cuerpos y las expresa de manera apropiada para servir de entrada a un programa de Elementos Finitos. Entre las aplicaciones prácticas se incluye la descripción de cavidades de radiación de calor entre un conjunto de cuerpos a varias temperaturas, para la solución del problema de la transferencia del calor.

El programa utiliza variables lógicas, para poder adoptar decisiones exactas en los casos en que es necesario determinar si, por ejemplo, tres puntos dados son colineales, o si dos cuerpos están en contacto o nó. Este informe describe el método utilizado y la preparación de los datos de entrada. Se incluyen algunos ejemplos simples.

CONTENTS

1. INTRODUCTION

The present work was motivated by a particular class of problems: the calculation of the temperature distribution in a pile of bodies which are being heated up or cooled down in a furnace /1/. Heat propagates by conduction inside the bodies and by radiation in the free spaces or enclosures left between the bodies. The problem is solved by the Finite Element method /2/ inside each body and by the Net Radiation method /3/ in each enclosure. The Net Radiation method requires that the boundary of each enclosure be approximated by elements, while the Finite Element method requires that the inside of each body be approximated by elements. Both the boundary elements and the body elements carry reference numbers. A complete description of such a physical system involves the coordinates of both the boundary and the body nodes, the element/node correspondence for each mesh of each of the two types, and the correspondence between each boundary mesh and the body mesh in the corresponding body. Preparing such a volume of data by hand proved to be very cumbersome for all but the simplest systems, and many human errors were made.

The present study was undertaken from a more general point of view: the automatic description and meshing of the boundaries of the free spaces left by a set of bodies, some or all of which may be in contact. Man/machine communication should be made as simple as possible.For this reason, the concept of repetition was introduced. When many bodies are alike, the user has to describe only one of them. As is often the case in many engineering applications, a complex system may be entirely defined by a few

numbers, and by a set of logical rules which indicate how the system is constructed. As an example, consider a pile of identical cylinders on a flat surface: the diameter of the cylinders is the only relevant magnitude, and then the rules which tell how each cylinder is set on top of others determine the system. The program works in exactly the same way: the user defines a few auxiliary points and then gives the logical rules for constructing the system. The program does everything else.

Another important aspect of this method is that, since the system is logically defined, exact decisions can be made concerning questions such as the interior and the exterior of a closed surface, the fact of a given surface being closed or not, and the collinearity of three given points. A geometrical treatment of such questions frequently leads to situations in which the final decision depends on rounding errors introduced by the computer, which is highly undesirable. Logical decisions, on the other hand, are exact.

The program we present here is general and may be used for many applications, one of which is the solution of heat transfer problems in piles of bodies.

2. PROGRAM METHOD

The techniques needed for the resolution of this problem are those of pure logic. For example, it is not possible to decide with certainty whether three points are collinear by the straightforward method, when using "REAL" variables, because of the rounding errors introduced in the calculation. Integer variables are not suitable because they do not normally describe the real world. The logical variables used are codes (which represent real variables) defined as follows:

As a first step in the data preparation, the user must assign code numbers from the set of prime numbers to all basic numeric values needed by the program. The program uses functions of these numeric values to describe the whole system. By default, any code C not explicitly defined by the user is taken to correspond to the numeric value $\sqrt{C}$. For example, if the user defines codes 5 and 7 to correspond to the values 9.25 and 3.142 respectively, then the program uses the following table of codes:

Code	Value
1	1
2	$\sqrt{2}$
3	$\sqrt{3}$
4	2
5	9.25
6	$\sqrt{6}$
7	3.142
8	$2\sqrt{2}$
9	3

etc.

A product of basic numeric values is represented by the product of their codes; since only prime numbers are used for the basic values, this representation is unique. Using the above table as an example, the code 147 (which factorises into 7x7x3) represents $(3.142)^2\sqrt{3}$. Exact multiples are represented by tuples of the form [m,c] where m is a multiplying factor and c is a code. In our example [6,10] represents $6 \times 9.25 \times \sqrt{2}$. Linked sets of such tuples are used to represent sums and differences of such expressions. For example, if the symbol → represents chaining within the program, then [2,2]→[5,1]→[-2,15] would represent $2\sqrt{2} + 5 - 2 \times 9.25 \times \sqrt{3}$. It can be seen that the basic variables must be defined to be the elementary "building blocks" from which the system is to be constructed without the need for division. For a circle of radius R which is to be divided into 12 elements equally spaced along its circumference, a basic variable of R/2 must be defined in order that the coordinates of the nodes may be expressed in this logical format. The program performs all operations on the logical codes, without needing to know the values of the corresponding real variables. Consequently, all decisions taken by the program are exact.

Man, using his mental and visual faculties, is in many ways superior to the machine when resolving this type of problem. He can see at once into how many zones the plane is divided by a given set of bodies, whereas the machine has to perform many operations of a complicated algorithm in order to arrive at the same result. This is because the human eye shows the relations between bodies in two dimensions, whereas the computer memory is

essentially one-dimensional. An algorithm has to be devised to enable the machine to overcome its "visual limitations". Then the machine becomes more efficient than the human being because of its speed and accuracy.

The algorithm used by this program is essentially one of algebra. The numeric values are treated as if they were unknowns, and each new expression evaluated is stored as a function of these unknowns. The tracing out of the zones is performed by trial and error, helped by "intelligent guess-work", in much the same way as a computer would find its way through a random maze. Tables are built to describe

i) The different types of zones found; congruent zones share the same entry in this table.

ii) The complete path traced out by the program, cross-referenced to the table of zone types. In a large system it is often found that many zones are congruent; these congruent zones may well be treated all alike by the Finite Element program and hence it is useful to separate the common subset of data in this way.

3. PROGRAM DETAILS

Data preparation, described in detail in section 4, consists of 3 stages;

i) definition of codes to represent the basic variables;

ii) definition of the auxiliary points, which are used as reference points for stage iii), e.g. the centre of a circle or the extremes of a line segment;

iii) definition of the bodies, e.g. for a circle, its centre, its radius, and possibly references to a table of material properties. When several identical bodies appear, a repetition factor may be specified.

In order to permit dynamic allocation of storage, all tables are stored in a single large array, but for the purpose of simplifying this report, names will be given to each of the tables. The n auxiliary points are stored in the table "TAP" with format of figure 1, where for example the first point has x

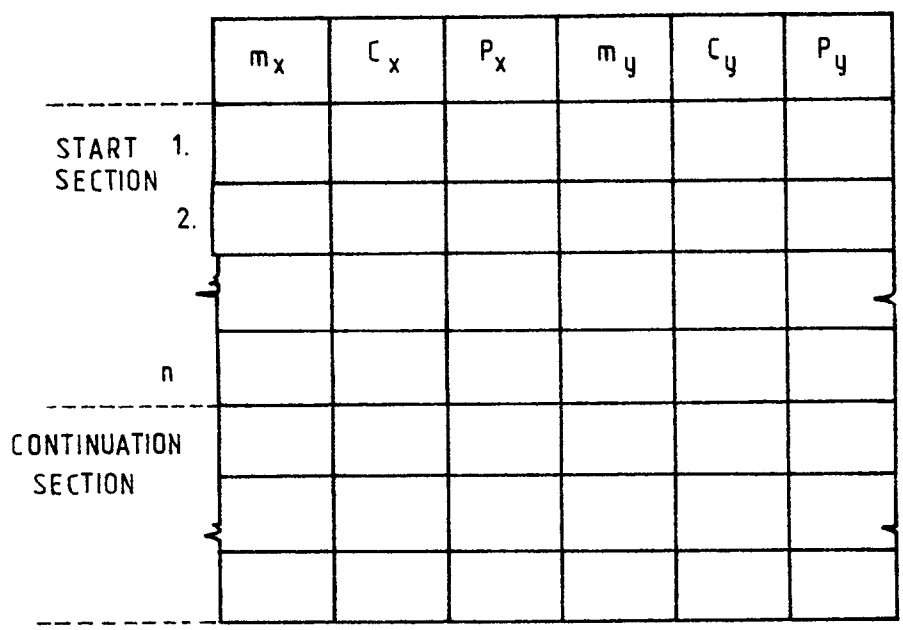


FIGURE 1

coordinate defined by the tuple $[m_x,c_x]$ plus further tuples found by following the pointer p_x. $p_x = 0$ indicates the end of the chain. The definition of the i^{th} point must begin in the i^{th} line of the table, and from there it can be chained to any other part of the table, whenever possible utilizing pieces of chain set up for other points. The first n lines, where the definition of all points must start, is called the "START" section of the table. The "CONTINUATION" section is used for the chained expression of which a copy cannot be found in the START section. For example, a point with x coordinate represented by [5,3]→[2,7] would have |5|3|→| in its corresponding positions in the START section, and might have |2|7|0| as an entry in the CONTINUATION section, the → serving to link the two expressions. Note that this table is prepared by the program from data supplied by the user in a much simpler format.

The program attempts to put all expressions in canonical form, to simplify whenever possible, and re-use memory which has been freed by a simplification procedure. For example

[1,27] would be converted to [3,3],

[2,5]→[4,5] to [6,5]

and [2,7]→[1,3] to [1,3]→[2,7]

(thus leaving the codes in numerical order).

The subroutine "SORTS" performs most of this work.

The user prepares data to describe the bodies ocupying the 2 dimensional space according to the instructions given in section 4. Henceforth, these bodies are referred to as "nets",

because each body is broken down by the program into a set of interconnected nodes. The program allocates consecutive node numbers, and builds up the table of nets "NET" with the format of figure 2.

NET No.	TYPE CODE	MATERIAL CODE	SIDES PER F.E.	C OR P_1	R OR P_2	FIRST NODE N°	LAST NODE N°

FIGURE 2

Certain types of net may be broken down by the program into subnets, thus occupying more than one line of the table. The types of net currently incorporated in the program are:

i) Type 0; a circular net of 12 nodes numbered in the anti-clockwise sense from the "3 o'clock" position, centre C, radius R. C is a pointer to the table TAP, and R is the code of the numeric value of the semi-radius of the circle.

ii) Type 2 with Material code M > 0; constitutes a straight line segment from P_1 to P_2 of material of negligible thickness with free space on both sides. P_1 and P_2 are pointers to the table TAP.

iii) Type 2 with M < 0; straight line boundary of the system; the space to the right of the segment, when going from P_1 to P_2, is considered to be outside the system.

iv) Type 2 with M = 0; M = 0 is the material code of a "mirror" used to define a boundary which is an axis of symmetry. The same convention applies, that the virtual mirror space is on the right of the segment P_1P_2.

Section 4 of this report shows several facilities which may be used for the automatic generation of the above nets; these facilities are used by the data input routine to produce nets and subnets with the above simplified format. For example, the "circular arc" is transparent to the program because it is translated into several straight line segments; the "repetition factor" does not appear in the table NET because the data is expanded to fill one line for each net. The "automatic node generation" facility causes nodes to be generated at each point where the line segment cuts a solid body, and the parts of the segment within the solid body to be eliminated. For example, the

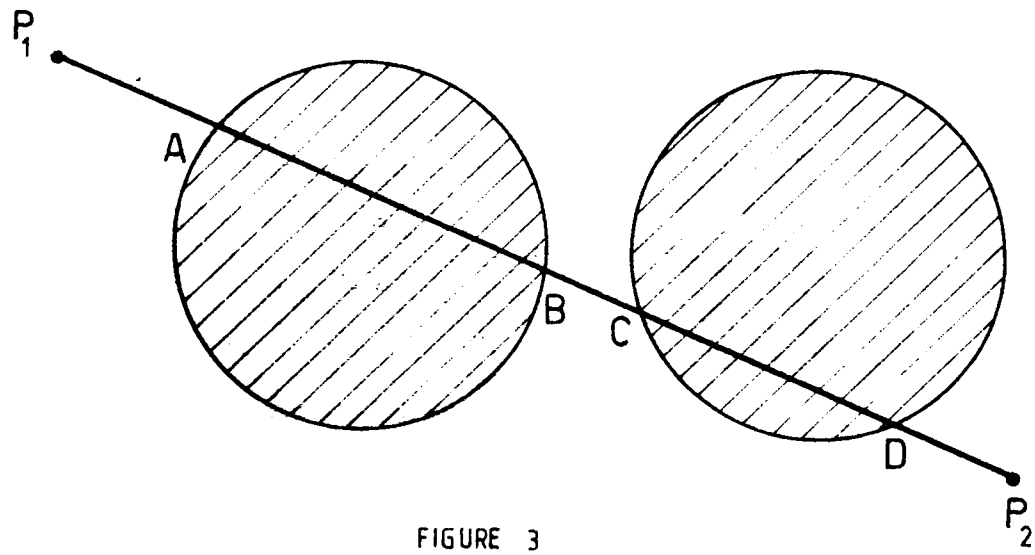


FIGURE 3

line of figure 3 from P_1 to P_2 with the "G switch" on would generate three subnets: from P_1 to A

from B to C

and from D to P_2.

This is particularly useful for layers of cylinders separated by thin flat plates.

The processing of the input data and the preparation of the tables TAP and NET is performed by the routine "SPADE1", making use of the following general purpose subroutines

i) MULT; multiplies together two logical expressions, and stores the result at the next available free space in TAP.

ii) ADSUB; adds or subtracts two logical expressions and stores the result at the next available free space in TAP.

iii) SORTS; simplifies and puts expressions into canonical form.

iv) COORD; evaluates the logical expressions which represent the coordinates of a given node.

Having processed, checked and expanded the input data, the program proceeds to its logical steps. Each logical step results in the generation of a new table. The table of coincident nodes, TCN, has the format of figure 4, thus indicating a one-to-one correspondence between pairs of coincident nodes. Such a superposition of nodes occurs at the point of contact of two or more nets. When three or more nets have a mutually common node,

NODE No.	NODE No.

FIGURE 4

TCN indicates this by an entry of the form shown in figure 5, which for example would indicate that the nodes α, β and γ are coincident and that the nets which contain α, β and γ are arranged in cyclic clockwise sense as in the diagram.

The cyclic ordering is necessary for the step of tracing out the free zones described later. The format of the table TCN is modified as follows, to permit a direct look-up procedure for each node (instead of having to search the table): one

α	β
β	γ
γ	α

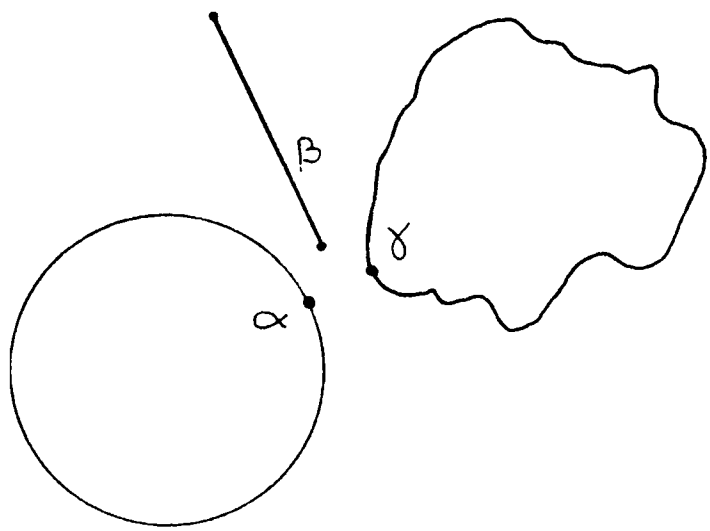
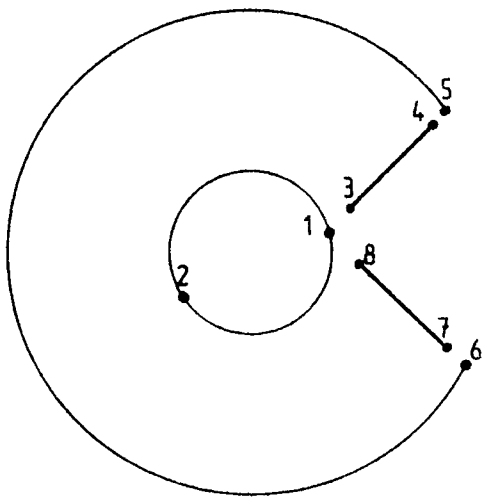


FIGURE 5

integer variable is allocated for each node of the system, and this variable contains the node number of a coincident node if such exists, or zero if there is no coincident node. The clockwise convention mentioned above is maintained. Then a negative prefix is assigned to any array element corresponding to a node which has been specified as the second end-point (P_2 on the data sheet) of a boundary net (m $\leq$ 0 on the data sheet). This prefix is later used to prevent tracing of a zone edge from P_2 to P_1. The simple example of figure 6 illustrates the final format of TCN.



	TCN
1	3
2	0
3	8
4	-5
5	4
6	-7
7	6
8	-1

FIGURE 6

TCN, which is prepared by the routine SPADE1, now contains the logical data needed for tracing out the free zones between bodies.

The main program traces out the zones as follows: possible starting points are considered to be those elements of TCN which are positive and non-zero; i.e. we always start the description of a zone at the point of intersection of two nets, but not on a boundary. Each free zone must have at least one such internal node. As the zone is traced out, the corresponding elements of TCN are given negative prefixes; they are made to look like boundary nodes so that the same zone is not traced again.

From a given starting node, the program traces a path along the surface of a net. For solid nets, of which the interior is considered external to the system, the net is followed in clockwise sense, thus tracing the free zone in counter-clockwise sense. At each node encountered, a coincident node is sought and, if found, the path is traced from one net to the other. When the "starting point" is again reached, a whole zone has been described. The process is repeated until all possible starting points have been exhausted.

The arrays NE and MP (described in section 5) are generated, as follows. NE, which contains entries for all zones, is buffered and written onto the output file. MP is stored in main memory, and is cross-referenced to an abbreviated version of NE (called NF, stored in main memory), and to the table of equivalent zones, "TEZ". Congruent zones need to share the same entries in MP, NF and TEZ; to this end, for each zone described by the program, new entries are made as if it were a new type of zone. Then, if it is found to be congruent to a previously described zone, the new entries are erased (by backspacing pointers) and

references are made to the previous entries. Thus the arrays MP, NF and TEZ are mantained relatively short; they serve the following proposes:

MP is as defined for the output interface in section 5, except that the pointer PP (not yet needed since the application dependent array P does not yet exist) is used as a pointer to the corresponding section of NF.

NF is equivalent to NE, but with a section for each type of zone only. A "sample" of NE is stored for each type of zone for later reference. This is because NE, having possibly been written onto a sequential device, is not readily available for reference. The pointer P of NE is replaced by a pointer P' of NF to the table TEZ.

TEZ describes each zone type and enables the program to decide which zones are congruent. When a whole zone has been described by the main program, the routine SPADE2 is called to make the corresponding entries in TEZ and then to compare this zone with all those zone types already described in TEZ. Its format is shown in figure 7.

NET TYPE	MATERIAL CODE	SIDES PER F.E.	LENGTH^2	A	POINTER

FIGURE 7

It has one line to describe each finite element, and the lines are chained together by means of the pointer in order to describe a whole zone. Each zone is described in counter-clockwise sense by a cyclic chain, thus facilitating the comparison of zones which by nature are cyclic. Length^2 is a cross reference to a logical data item which describes the square of the length of the finite element, and A similarly indicates twice the area of the triangle formed by this finite element, the previous finite element, and a chord joining the first node of the previous element to the second node of this element. A is calculated from the determinant $\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix}$, and consequently its sign indicates a left or right-hand sense in the description of the zone. The geometry of the zone is thus uniquely defined, independently of the orientation and without reference to angles. It is interesting to note the similarity between this method of defining the shape of a zone, and the ideas of distance geometry /4/. Length and areas are intrinsically easier to calculate than angles because the only operations involved are those of addition, subtraction and multiplication. The logical variables representing all lengths and areas are stored in the extension of the table of auxiliary points (TAP) in the same format as the coordinates of points. Whenever a new length or area is created, it is put into canonical form and compared with previous entries in the table to see whether a previous entry can be used for the representation. Consequently, comparison of

two zones is reduced to a direct comparison of the entries in the table TEZ.

When two zones are found to be congruent, the zone most recently described is eliminated from TEZ, and from MP; the elements of NE are cycled to match those of the existing entry in TEZ, and all cross references are made to this previous entry.

The routine SPADE2 completes the description of the zone as follows: For a new type of zone, NF is created, and MP has the "sides per finite element" filled in. For all zones NE has the "element numbers" filled in, has mirrors deleted (they figure as open sides in the final representation) and has the "reduction" process performed; i.e. reduction of a whole net to a single finite element as described in section 4.

Figure 8 shows the cross references which exist between the various tables during the process of tracing out the zones. Note that TEZ also contains a pointer, back to MP, in order to facilitate the finding of the relevant section of MP after a congruent zone has been found.

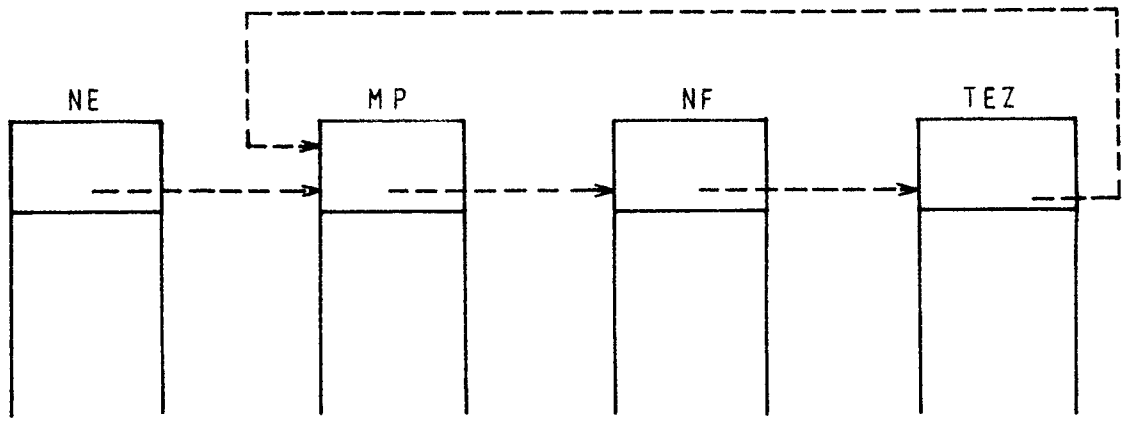


FIGURE 8

Many sections of NE may point to one section of MP, but there is a one-to-one correspondence between sections of MP, NF and TEZ.

The final stage of the program is application dependent. For example it may be used to calculate the radiation matrix /1/ for each type of zone, in which case some further logical calculations are needed in order to prepare data for the routine 'MATRAD'. We need to decide, logically, whether any two sides are in line with each other; and if there is a mirror in the system, to decide whether a side is in line with the image of itself or of another side. Such logical decisions may be made by the program, using its basic routines for addition, subtraction, multiplication and simplification of logical expresions.

4. DATA PREPARATION

4.1. Code numbers

Before attempting to prepare data, the user should have a clear understanding of section 2 of this report, in particular the use of code numbers to represent all basic numerical values of the system. A data preparation form is included in Appendix D.

It must be emphasized that the basic values needed by the program must be anticipated and supplied by the user. For example, a line segment of length 3.25 units, from the origin, in the direction (1,1), to be represented by 6 nodes (i.e. 5 elements) would require a basic variable of

$$3.25 \times \frac{1}{\sqrt{2}} \times \frac{1}{5} = 0.325\ \sqrt{2}$$

Since $\sqrt{2}$ is implicitly represented by the code 2, the basic variable 0.325 could be represented by code 7 for example. Then the length of the segment would be represented by the tuple [5,14] in both x and y directions, interpreted as $5 \times 0.325 \times \sqrt{2}$ according to the rules defined in section 2.

A circular net, centre the origin, radius 26.4 units, with 12 nodes, needs a basic variable of 13.2 to be defined, with code 11 for example (Note that the codes are prime numbers, and that we usually reserve 2 and 3 to represent $\sqrt{2}$ and $\sqrt{3}$ respectively). Then its "3 o'clock" node would have x coordinate [2,11] and y coordinate [0,0] allocated by the program. Its "2 o'clock" node would have x coordinate [1,33] and y coordinate [1,11]. It can be seen that the program is most useful when the system is repetitive in nature and can be broken down into few basic numerical lengths. The highest code number allowed is 47.

4.2. Material properties

This data table is application dependent; it consists of a reference no.M ($M \leq 10$) for each material and a list of properties. For the example of the calculation of radiation enclosures /1/, the properties are emissivity and initial temperature; two nets with different initial temperatures are regarded as made of different "material".

From this point onwards, all data are specified in terms of whole numbers; fractional quantities are defined by cross reference to the above two tables.

4.3. Auxiliary Points

Auxiliary points must be defined to enable the system of nets to be specified by cross reference to this table. Such points as centres of circles and extremes of line segments are defined and numbered in increasing sequential order. However, when the end of a line segment coincides with a node of another net, it will be seen that there is no need to define this point explicitly; it can be referred to as the "i^{th} node of the j^{th} net", already previously defined. For example, in figure 9

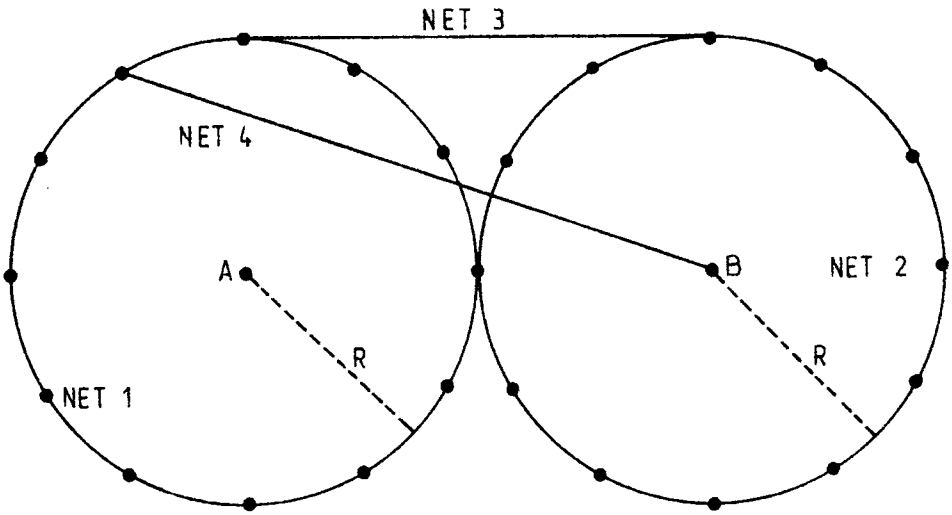


FIGURE 9

net 1 is a circle, of 12 nodes, centre A

net 2 is a circle, of 12 nodes, centre B

net 3 is a line segment, linking node 4 of net 1 to node 4 of net 2

net 4 is a line segment, linking node 5 of net 1 to the auxiliary point B.

In this simple case, only A and B need be defined as auxiliary points (and R/2 is the only numeric value to be coded).

The repetition factor N is useful when a number of collinear, equidistant auxiliary points have to be defined. The entries in the table are as follows:

P = This line refers to an already defined point P in form of "offset from point P".

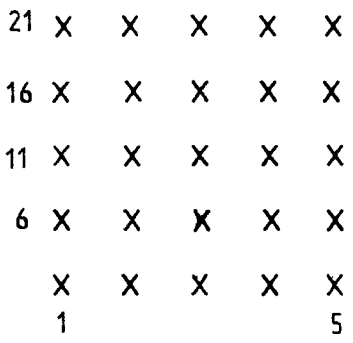
P = 0 means that this line defines absolute values of coordinates.

$[m_x, c_x]$ = tuple defining the x-coordinate of the point, relative to point P (or the absolute value when P=0).

$[m_y, c_y]$ = tuple defining the y-coordinate (similarly).

N = repetition factor. N points are defined; this is equivalent to writing out N lines of the table, all identical, except that the value of P is taken to be one higher for each successive line. Omission of the value of N, or putting N=0 or 1 defines a single point. N must be omitted when P=0.

As an example, figure 10 represents a uniform grid of points, which would be defined as shown.



P	m_x	C_x	m_y	C_y	N
0	0	0	0	0	0
1	1	7	0	0	4
1	0	0	1	7	20
-1					

FIGURE 10

Line 1 defines point 1 at the origin. Line 2 defines points 2 to 5 each successively v units further "east" where v is the value corresponding to code 7. Line 3 defines points 6 to 25 each v units "north" from a set of points starting with point 1. Note the useful recurrence relation used to define such a uniform figure. Line 4 indicates the end of the table.

4.4. Program switches

The four data items MAXD, MIND, MPAK, and MNUM may be supplied by the user in order to help the computer to resolve the problem more quickly. For each value which is not known or not applicable, zero should be supplied. These data items refer to systems containing many circular nets in close contact.

MAXD: In section 4.5, nets will be defined and numbered. MAXD is the maximum difference which exists between the numbers of two contiguous circular nets. In order to improve the efficiency of the program one should try to make MAXD as small as possible.

MIND: When circular nets are placed in a uniform pattern, and numbered sequentially by rows, any two contiguous nets either have consecutive numbers (if they are in the same row) or numbers differing by at least some integer (MIND) which is of the order of magnitude of the number of nets per row. i.e. "MIND" is the minimum difference between the numbers of any pair of contiguous nets, not counting a difference of value one; it should be made as large as possible.

MPAK: Is a code representing the method of packing of circles. MPAK =3 means circles arranged in "square array" as in figure 11a. MPAK=2 means "close-packed" circles as in figure 11b. MPAK=0 means that the circles are not packed according to any fixed rule.

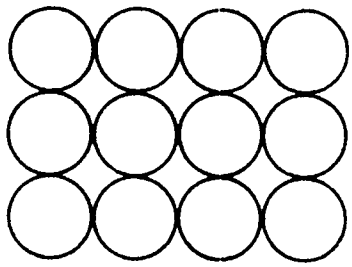


FIGURE 11a

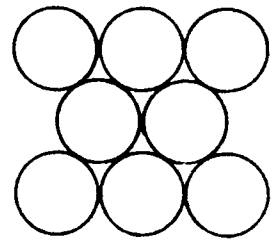


FIGURE 11b

MNUM: Is a code representing the method of numbering of circles. MNUM=1 means "numbering from left to right by rows starting with the bottom row". MNUM=0 otherwise.

The data item "PLOT" is a program switch indicating whether the system is to be drawn by the plotter. PLOT=1 causes the data to be plotted; this should be the normal choice, in order to verify the input data by visual means. PLOT=0 may be used when, for example, the plotter is not available.

4.5. Circular Nets

Currently included in the program is the treatment of the following types of net:

i) Circular nets with 12 nodes (type 0)

ii) Line segments of 2 or more nodes (type 2)

iii) Circular arcs of 2 nodes (type 2 with G=0, H≠0)

iv) Rectangular plates, with geometrical thickness, but in which an element on one face is regarded as being the same element as that in the corresponding position on the opposite face. This type of net is useful when it is known that opposite faces are always at the same condition (e.g. temperature), so that they can be represented by the same finite element, but when geometrical thickness is required in order to represent the true relationship between nets.

v) Mirrors, which are straight line boundaries representing axes of symmetry. Symmetric systems may thus be broken down into simpler subsystems.

Type 0 circular nets are defined first, and then all other types of nets are specified in any order. For all types, the first three parameters are as follows:

T = type of net = 0 for circular net
= 2 for all others.

M = material code, which is a cross reference to the material properties table described in section 4.2.

S = number of sides per finite element (it is often useful to divide finite elements into "sides" in order to obtain greater geometrical accuracy). See reference /1/.

For the type 0 net, the remaining parameters are as follows:

C = no. of the auxiliary point which corresponds to the centre of the circle (see 4.3).

R = code corresponding to the semi-radius of the circle (see 4.1). Note that a tuple can always be written as a simple code by

incorporating the multiplying factor in the code itself. Thus [3,7] ≡ [1,63] (because a multiplying factor of 3 is equivalent to a code of 9), and this may then be written as a code of 63 without need for a multiplying factor.

E = no. of finite elements on this net (at the time of writing, E = 12 is the only permissable value).

N = repetition factor. If given, N ≥ 2 specifies the number of circular nets defined by this data line; all nets are identical except for the value of C which is one higher for each successive net, starting with the given value of C.

Q = equivalence switch. Q=0 has no effect. Q=1 causes the program to consider each circular net as a single finite element. The zone shapes are evaluated geometrically as though each net had E finite elements, and then the results are reduced by lumping to a single finite element for each circular net which has the parameter Q=1. Note that the definition of the parameter Q for straight line segments is slightly different; for circular nets, the reduction is made net by net, but for line segments the equivalence is made between all segments defined on the same data line (Q=1) or between pairs of segments on two data lines (Q=-1, or -2). See section 4.6.

A typical data line might be that of 25 nets, centred respectively on each of the 25 auxiliary points defined in the example of section 4.2;

T = 0

M = 1, for example, referring to the 1st. material.

S = 2, for example, giving a 24 sided regular polygon.

C = 1, indicating the 1st.auxiliary point.

R = 63 (= 3^2 x 7) indicating a semiradius of 3 times the value of which the code is 7.

E = 12, indicating 12 finite elements per net.

N = 25 to give 25 nets altogether, all congruent, and with centres on the auxiliary points 1 to 25.

4.6. Line segments and circular arcs

For the type 2 line segment the parameters are as follows:

M = The material code M is as for the circular net; except that M=0 indicates that the surface is to be regarded as a mirror, defining a boundary which is an axis of symmetry; also a negative prefix on the value of M is used to indicate that this surface is a boundary (external or internal) of the system. Conventionally, the system is regarded as being on the left of the segment when passing from A to B (see below). Hence external boundaries must be described in the anticlockwise sense, and internal boundaries in the clockwise sense.

S = no. of sides per finite element (as for type 0).

A = Indicates one extreme of the line segment; it may be either (i) a reference to a node on a net already defined, in the form A = 100m+n ($1 \leq n \leq 99$), where m is the net no. and n is

the node no. relative to this net. Or (ii) a reference to an auxiliary point, with a negative prefix.

B = As A for the other extreme of the segment. A typical example might be A=304, B=-15 indicating a line joining the fourth node on the 3rd. net to the auxiliary point no.15.

G = Is the "generation" flag. G=0 has no effect; G=1 causes the program to generate nodes at all points, on this line segment, which coincide with nodes on previously defined nets. A restriction of the program (which in practice is not serious) is that these previously defined nets must have been defined in the same order as they are found when passing from A to B for this segment. The net AB is broken down by the program into a number of subnets joining node to node, and subnets which lie in zones external to the system (such as the insides of circular nets) are eliminated. This facility is useful for the specification of a series of line segments, such as those of figure 12, simplifying the data preparation.

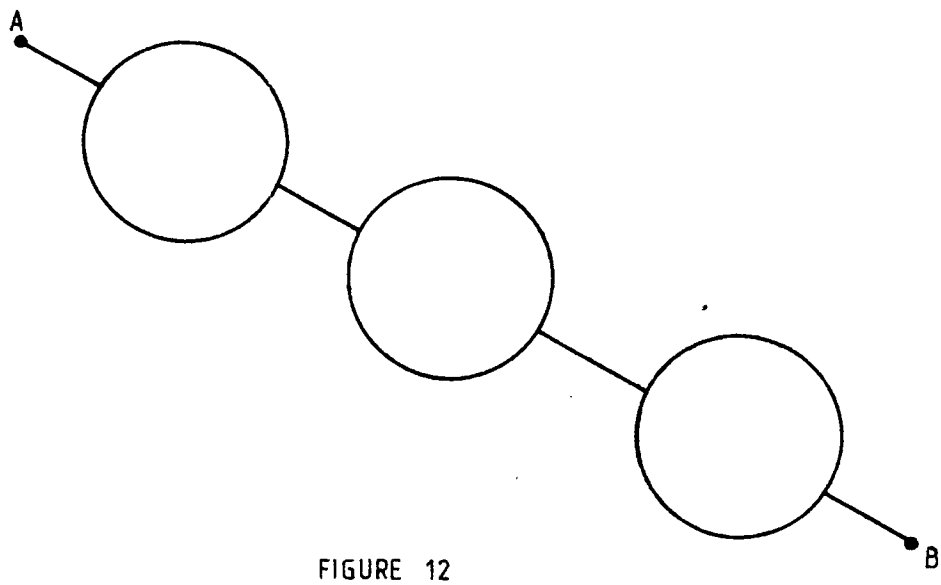


FIGURE 12

G = -1 has the same effect as G=1, except that the previously defined nets must have been specified in the reverse order, i.e. as found when passing from B to A.

H = (G≠0) It often happens that the above line segment intersects each of the nets in a given node (relative to each net). H is this node no., if known. H=0 yields the same result but needs more computation time. For example, in figure 13 H=12 would make the program more efficient.

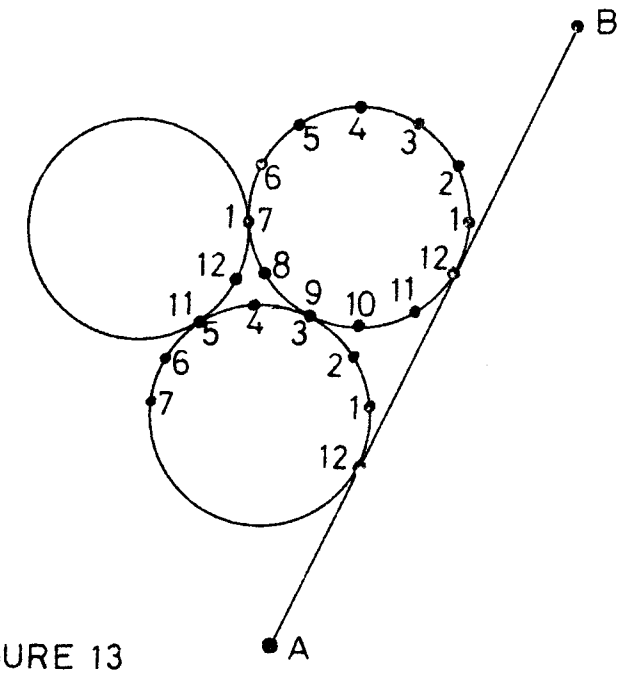


FIGURE 13

H = (G=0) An alternative interpretation of H (when G=0) is that the segment is to be regarded as curved with centre of curvature at the auxiliary point H. Is should be noted that, although an improved geometrical representation is obtained, this arc consists of only one finite element.

Q = The optional parameter Q may be used for the specification of

"equivalent nets". It is often useful to be able to define several geometrically distinct segments as belonging to the same finite element. Obvious examples are the wedge and the thick plate of figure 14; BC may be considered as belonging to the same element as FG, but for a correct geometrical interpretation of the free zones between bodies, the wedge cannot be given zero thickness.

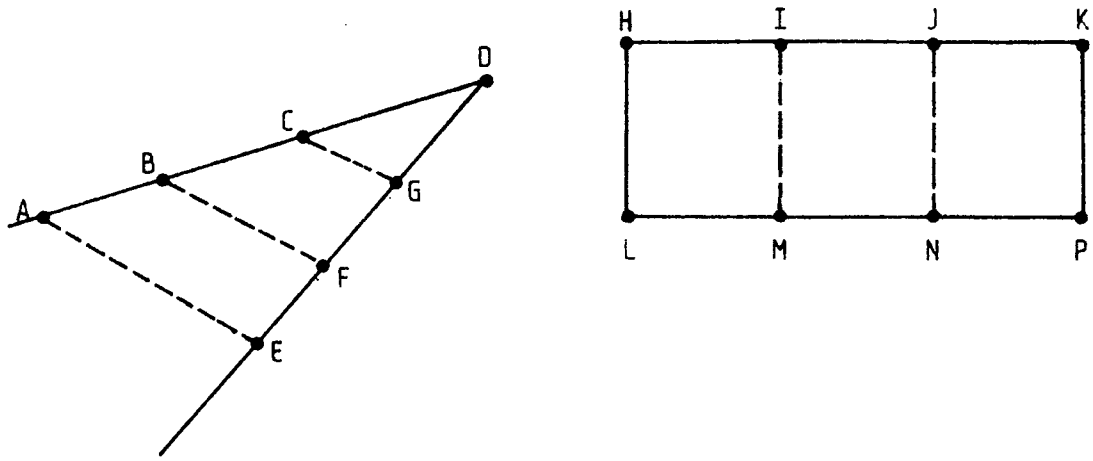


FIGURE 14

The codes for Q are as follows:

Q = 1 means that all segments defined on this data line are equivalent (several segments may be defined in the same line by putting G≠0).

In the example of figure 15, taken from a heat transfer problem, if AE is a perfect conductor, all segments could be represented by a single finite element, but geometrically

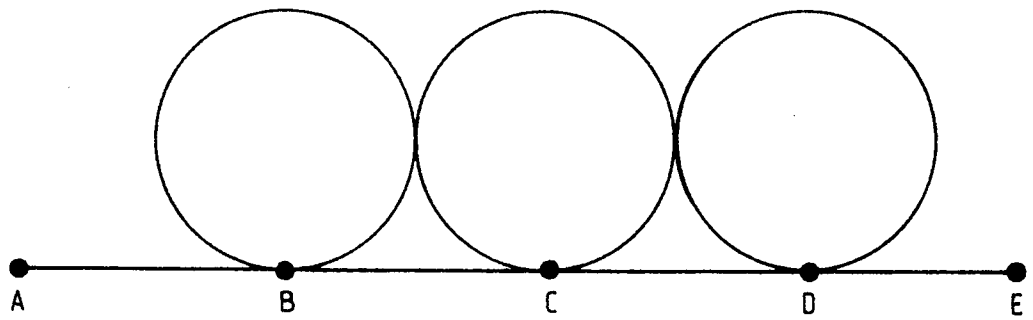


FIGURE 15

the distinct subnets AB, BC, CD, and DE are needed in order to complete the radiation zones between bodies.

Q = -1 means that the segments defined on this data line are equivalent by pairs with those of the previous data line, taken in reverse order. For example, for the wedge of figure 14 we could specify:

from A to D, G=1, H=0, Q=0

from D to E, G=-1, H=0, Q=-1

implying that AB ≡ EF, BC ≡ FG and CD ≡ GD.

Q = -2; as Q=-1, except that reference is made to the data line two lines above. For the thick plate of figure 14, we could write:

from H to K; G=1, H=0, Q=0

from K to P, G=0, H=0, Q=-1

from P to L, G=-1, H=0, Q=-2

from L to H, G=0, H=0, Q=-1, thus splitting the plate into three finite elements, as indicated by the dotted lines. Furthermore, use of a negative prefix on the material code would exclude the inside of the plate from the system.

4.7. List of nets to be monitored

This data list is passed directly to the finite element program, to indicate which nets are of interest for the printing of results.

5. PREPROCESSOR OUTPUT INTERFACE (RELEASE 1)

The output interface consists of the following variables, in this order:

NE an array defining the path to be followed in order to describe the edges of all the zones (enclosures).

LNE the length of NE.

LMP the length of MP (see below)

LP* the length of P (see below)

NNTZ the number of types of zone

NNZZ the number of zones

NNND the number of nets

NNIMP the length of NIMP (see below)

NTUB the number of circular nets

T1*initial temperature of material no.1

T2*initial temperature of material no.2

NIMP array of nets to be monitored (see input data)

MP array of properties of each type of zone

NEN array of number of finite elements originally specified per net (before reduction); length of array = NNND.

P* array of heat radiation matrices for each type of zone.

T* array of initial temperatures for each net.

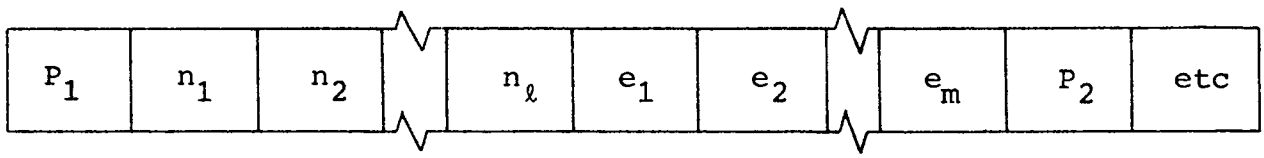
The data is written onto a sequential (or random access) device, blocked according to the user's convenience. Each integer variable (FORTRAN convention) occupies 2 bytes and each real variable occupies 4 bytes. All arrays start on a new block, and the

---

* Application dependant /1/.

single variables (from LNE until T2) are all written on the same block. The array NE is terminated by at least one zero, and its length is given for verification purposes (NE, which may be a large array, is written onto the file as it is generated, and consequently its length is not known until after it has been written). All other arrays are controlled by their lengths.

NE and MP are the two application independent arrays which describe the whole system. Basically, NE runs over the edges of all zones, and is cross referenced to MP which contains data relative to each type of zone. NE has the following format:



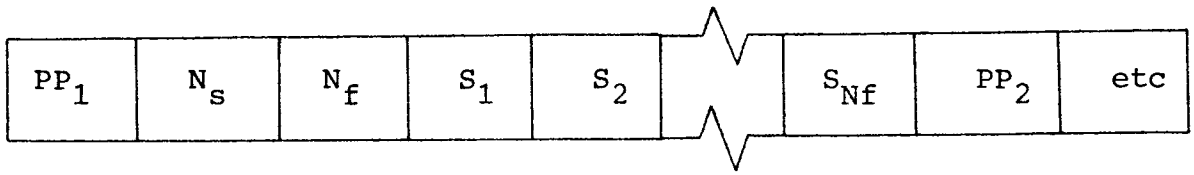
where P_1 to e_m represents one zone.

$\underline{P}$ is a pointer to the corresponding section of the array MP.

$\underline{n_1 \ldots n_\ell}$ are the node numbers which define this zone, sequentially in anticlockwise sense.Any node which is coincident with the following node in the list is given a negative prefix.

$\underline{e_1 \ldots e_m}$ are the numbers of the finite elements defining the zone, with e_1 joining n_1 to n_2 and so on. Note that there are less elements than nodes because two coincident nodes have no element joining them.

MP has the following format:



where PP, to S_{Nf} refers to one type of zone. Two zones are congruent if they are identical in all respects (including material properties) except for position and orientation; such zones share a common entry in the array MP.

PP is a pointer to the array P of application dependent properties of the type of zone. For the example of heat radiation enclosures, P is the radiation matrix.

N_S is the total number of sides for this type of zone.

N_f is the number of finite elements for this type of zone.

$S_1 \ldots S_{Nf}$ are the numbers of sides for each element respectively; S_i corresponds to e_i in NE.

Note that N_f controls the length of the entries S_i in the array MP, and also controls the n_i and e_i of the array NE. It is not possible to interpret NE without reference to MP.

Examples are given in appendices A and B. Note that, in these examples, because of the reduction of the circular nets to single finite elements, there is much repetition of element numbers and node numbers.

APPENDIX A - A simple example

The following example illustrates the basic ideas of data preparation. The example is so elementary that it would be easier to prepare the output of the preprocessor than the input data, but it is included in order to help in understanding the preprocessor program. The example is taken from a problem of heat radiation /1/, and consists of two long cylinders placed in contact with two plates as shown in figure 16.

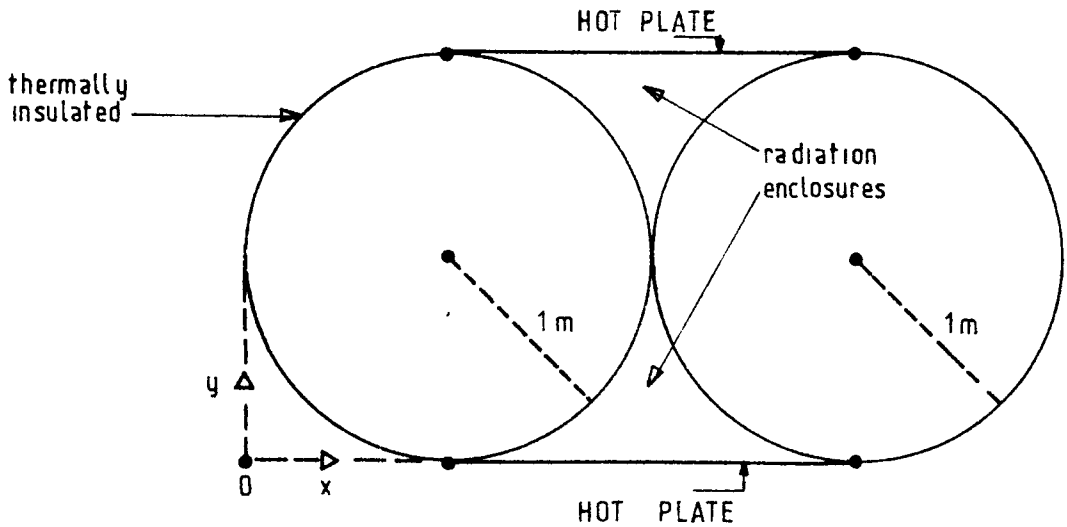


FIGURE 16

The exposed half of each cylinder is thermally insulated and the plates are maintained at constant temperature of 500°C. The cylinders are assumed to be isothermal and heated purely by radiation, since contact with the plates is poor. It is required to plot the temperature of the cylinders against time.

The only basic numeric value required is the semi-radius of the cylinders, given as 0.5m with a code of 7. The two materials

defined are those of the cylinders and of the plates respectively and the two auxiliary points are the centres of the cylinders. The two plates are defined as external boundaries (M < 0) and the program deduces that the exposed surfaces of the cylinders are outside the "system".

SPADE DATA PREPARATION FORM (SHEET 1)

HEADING:

SIMPLE EXAMPLE. APPENDIX A

CODES:

CODE C	VALUE V
7	0.5
-1	

MATERIAL PROPERTIES

	EMISSIVITY	INITIAL TEMP. °C
1	0.200	20.0
2	1.000	500.0
-1		

AUXILIARY POINTS:

	RELATIVE TO: P	Mx	Cx	My	Cy	REPETITION N
1	0	2	7	2	7	0
2	1	4	7	0	0	0
3						
4						
5						
-1						

NOTE: PUNCH ALL DATA ENCLOSED IN BOXES.

SPADE DATA PREPARATION FORM (SHEET 2)

PROGRAM CONTROL:

MAX D	MIN D	MPAK	MNUM	PLOT
0	0	0	0	1

CIRCULAR NETS:

TYPE T	MATERIAL M	SIDES/FE S	CENTRE C	RADIUS R	ELEMENTS E	REPETITION N	EQUIVALENCE, Q
Ø	1	2	1	7	12	2	1

LINE SEGMENTS:

TYPE T	MATERIAL M	SIDES S	END - POINTS A	END - POINTS B	GENER. N/o G	H*	EQUIVALENCE, Q
2	-2	2	110	210	0	0	0
2	-2	2	204	104	0	0	0
-1							

*H = NODE No FOR G ≠ 0
ARC CENTRE FOR G = 0

NETS TO BE MONITORED:

1							
-1							

```
SIMPLE EXAMPLE.  APPENDIX A.

NETS AND NODE NOS...
  3  1  2     1     7   1  12   0   0        0  1  2     2     7  13  24   0   0        2 -2  2   110   210  25  26   C   C
  2 -2  2   234   134  27  28   0   0

COINCIDENT NODES...
  19     3     3    28     0     0     0     0     0   25     0     0     0     0     0   27     0     0     1     0     0   26     C     C    10
 -22    15    -4

NE... NODES AND ELEMENTS...
   1     1     1     1    -1     3    -3     2     2     2    -2     1     1     1     3     2     2     2     1     2     2     2    -2     4    -4     1     1     1    -1     2
   2     2     4     1     1     1

STORAGE ALLOCATION...
AUX. POINTS          1     10
NETS                13     43
NEN                 53     53
NEE                 53     98
MP                3999   4008
NE                6999   4599
LEZ               5003   5014
WORK-SPACE        5999   6060
START PTS.           0  10001
COIN. NODES       3672  13000

EQUIVALENT ZONES...
     3     0     1     2     7 5007     1          1     2    -2     2 6047 5014     0          3     0     1     2     7 5000     0

NE...ABBREVIATED VERSION OF NE...
 5000     1    12    11   -10    25   -26    22    21   -20   -19

ZONE TYPE 1
    NA    NB   LEN          S                V              EMIS
     1     2     1     0.20000E 01     0.10000E 01     0.20000E 00
     2     3     2     0.19659E 01     0.74118E 00     0.20000E 00
     3     4     3     0.18660E 01     0.50000E 00     0.20000E 00
     4     5     4     0.17071E 01     0.29289E 00     0.20000E 00
     5     6     5     0.15000E 01     0.13398E 00     0.20000E 00
     6     7     6     0.12588E 01     0.34074E-01     0.20000E 00
     7     8     7     0.10000E 01     0.0             0.10000E 01
     8     9     7     0.20000E 01     0.0             0.10000E 01
     9    10     9     0.30000E 01     0.0             0.20000E 00
    10    11    10     0.27412E 01     0.34074E-01     0.20000E 00
    11    12    11     0.25000E 01     0.13398E 00     0.20000E 00
    12    13    12     0.22929E 01     0.29289E 00     0.20000E 00
    13    14    13     0.21340E 01     0.50000E 00     0.20000E 00
    14     1    14     0.20341E 01     0.74118E 00     0.20000E 00

MP...DATA FOR EACH TYPE OF ZONE...
     1    14     7     2     2     2     2     2     2     2

LENGTH OF NE =    35          NO. OF REAL NETS =    4          NO. OF ZONES   =    2
LENGTH OF MP =    13          NO. OF TUBES     =    2          NO. OF TYPES   =    1
LENGTH OF  P =   135          NETS MONITORED   =    1          NO. OF MIRRORS =    0

NETS MONITORED...
     1

FINITE ELEMENTS PER NET...
     5     5     1     1
) DAZIT
/R
TIEMPO 00 03 22      3      0
```

APPENDIX B - Example of mirrors and "generation"

This example is included to illustrate some of the more complicated facilities of the program. The system is small, and does not exhibit the power of the program for resolving systems consisting of several hundred nets, but it shows how to prepare data in the most efficient manner, taking advantage of axes of symmetry and using repetition factors and automatic generation of subnets. The full system is shown in figure 17, and the subsystem modelled is shown in figure 18 (because of the 6 axes of symmetry, it is sufficient to perform the calculation for 1/12 of the whole system).

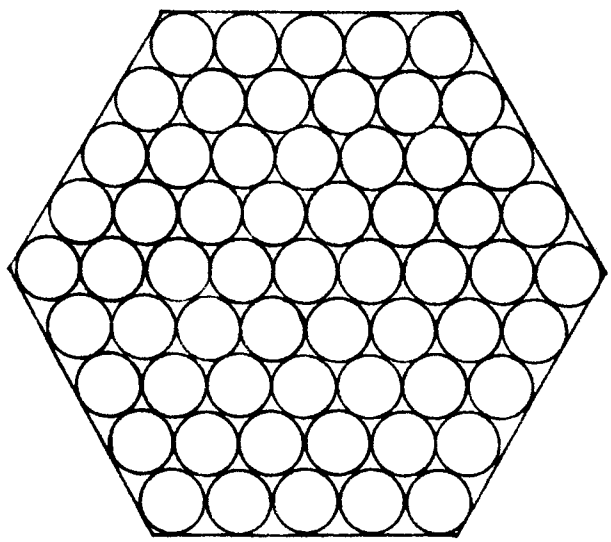


FIGURE 17

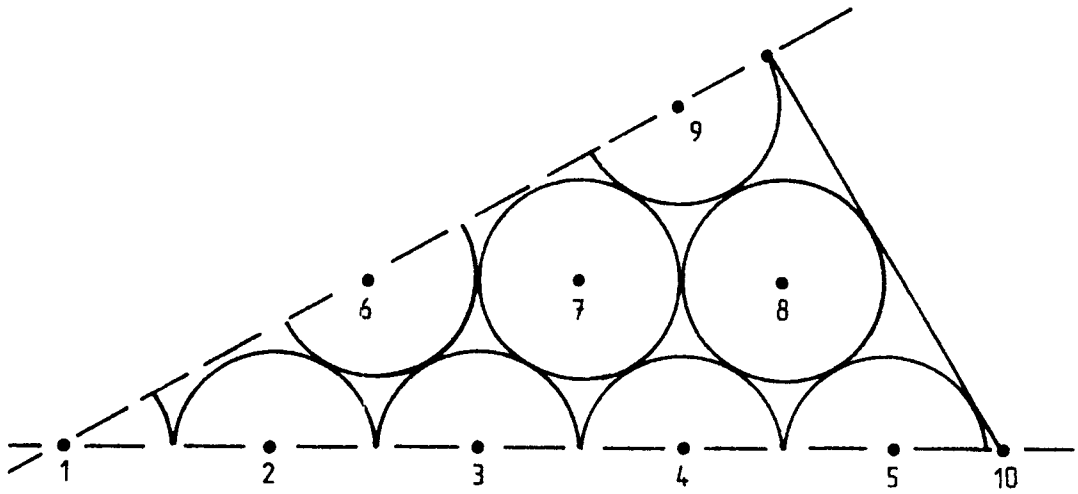


FIGURE 18

The only basic variable needed is 1/12 of the radius of each cylinder, which is assigned a code of 7 and a value of 0.423. It is easy to verify that the semi-radius is not sufficiently small to permit the cooordinates of all rodes to be expressed as integral multiples, but that R/12 satisfies this condition. The auxiliary points are marked in the diagram. MAXD has a value of 4, which speeds up the program slightly, but MIND=2 does not help and is omitted. The cylinders are close packed, and are numbered by rows left to right, giving MPAK=2, MNUM=1. Note that all 9 cylinders are defined in one data line and that the code for the semi-radius is $252 = 6^2 \times 7$, which represents a semi-radius of 6 x 0.423. Note also the way in which "generation" is used to define the boundaries efficiently: the circular nets are defined as whole circles and are then cut by line segments with

$G \neq 0$, which causes the circular arcs outside the system to be discarded, and line segments to be generated as subnets between circle and circle.

The external boundary consists of a combination of mirrors (which by nature define boundaries) and line segments with $M < 0$. The parameter H is given the value 2 for the external line segment in order to indicate that this segment intersects circles only at their 2nd node; but $H=0$ for the mirrors because they cut circles at various node numbers.

SPADE DATA PREPARATION FORM (SHEET 1)

HEADING: THEORETICAL OVEN. 61 TUBES.

CODES:

CODE C	VALUE V
7	0.4213
-1	

MATERIAL PROPERTIES

	EMISSIVITY	INITIAL TEMP. °C
1	0.2	20.0
2	1.0	500.0
-1		

AUXILIARY POINTS:

	RELATIVE TO: P	Mx	Cx	My	Cy	REPETITION N
1	0	12	7	0	0	0
2	1	24	7	0	0	4
3	2	12	7	12	21	3
4	7	12	7	12	21	0
5	5	8	21			
-1						

NOTE: PUNCH ALL DATA ENCLOSED IN BOXES.

SPADE DATA PREPARATION FORM (SHEET?)

PROGRAM CONTROL

MAX D	MIN D	MPAK	MNUM	PLOT
4	0	2	1	1

CIRCULAR NETS

TYPE T	MATERIAL M	SIDES/FE S	CENTRE C	RADIUS R	ELEMENTS E	REPETITION N	EQUIVALENCE, Q
0	1	2	1	252	12	9	1

LINE SEGMENTS

TYPE T	MATERIAL M	SIDES S	END - POINTS A	END - POINTS B	GENER.N/o G	H*	EQUIVALENCE, Q
2	-2	1	-10	902	1	2	0
2	0	1	-9	-1	-1	0	0
2	0	1	-1	-10	1	0	0
-1							

*H = NODE No FOR G ≠ 0
ARC CENTRE FOR G = 0

NETS TO BE MONITORED

1	2	3	4	5			
-1							

```
THEORETICAL OVEN.  61 TUBES.

NETS AND NODE NOS...
   3   1   2     1    252     1    12     0     0        0   1   2     2    252    13    24     0     0        0   1   2     3    252    25    36     0     0
   3   1   2     6    252    37    48     0     0        0   1   2     5    252    49    60     0     0        0   1   2     6    252    61    72     0     0
   3   1   2     7    252    73    84     0     0        0   1   2     8    252    85    96     0     0        0   1   2     9    252    97   108     0     0
   2  -2   1   -10    532   109   110   -11     0        2  -2   1   502    802   111   112   -11     0        2  -2   1   802    502   113   114   -11     0
   2  -2   1   302    -11   115   115   -11     0        2   0   1   908    705   117   118   -12     0        2   0   1   705    602   116   120   -13     0
   2   3   1   503    235   121   122   -12     0        2   0   1   205    102   123   124   -12     0        2   0   1   101    207   125   125   -13     0
   2   3   1   201    307   127   127   -13     0        2   0   1   301    407   129   129   -13     0        2   0   1   401    507   131   131   -13     0

COINCIDENT NODES...
  -13   126     3     0     0     0     0     0     0     0     0     0   -31     0    69     0   123     0     1     0     0     0     0     0   -43
    3    31     3    71     0    13     0     0     0     0     0   -55     0    93     0    83     0    25     0     0     0     0     0  -123   111
    3     3    35     3    37     0     0     0     0     0    79   120     0     0     0     0     3  -121    15     0    29     0    51     0   105
    3   113     3    61     0    27     0    41     0     0   113     0     0   107     0    73     0    39     0    53     0     0  -114     0     0
    3     3     3  -117    75     0    89     0   134   -50   110   -86   112   -98   115     0   104   -77   118   -62    68   -17   122    -2   125
    3   127     3   129     0   131     0    49  -105

NE... NODES AND ELEMENTS...
   1     1    -1     2     2    -2     1     2     2     8     2     2    -2     3     3    -3     6     6    -6     2     2     3     3    -6     6    17     2     2    -2     6
  -3     2     2     5     8     3     3    -3     4     4    -4     7     7    -7     3     3     4     4     7     7     3     3    -3     7     7    -7     6     6    -6
   3     3     7     7     6     6     8     4     4    -4     5     5    -5     8     8    -8     4     4     5     5     8     8     6     4     4    -4     6     6    -6     7
   7    -7     6     6     8     8     7     7    24    10   -10     5    -5    10     5     3     5     5     5    -5    11   -11     5     8     8    -8     5     5    -5    11
   3     3     3     1     6    -5     7     7    -7     6     7     7     8     7     7    -7     8     8    -8     9     9    -9     7     7     8     8     9     9    17     7
   7    -7     9    -9     7     7     9

EQUIVALENT ZONES...
     1     3     1     2   252  5007     1        2     0     1     2   252  5014     0        1     2     3     1  6071  5000     0
     2     3     1     2   252  5028     8        2     0     1     2   252  5035     0        1     2     3     2   252  5021     0
     2     3     1     2   252  5049    17        1     0     1     2   252  5056     0        1     2     3     1  6071  5042     0
     1     2    -3     1  6161  5070    24        1     0     1     2   252  5077     0        1     2     3     1  6122  5063     0
     3     3     1     2   252  5091    30        1     2    -2     1  5221  5098     0        3     3     1     2   252  5084     0

NE...ABBREVIATED VERSION OF NE...
  5333     2    -1    13    18   -17   123  -124  5221    15    14   -13    31    30   -29    71    72   -69  5042    17    16   -15    65   -66   121
  -122  5053   139  -113    53   -49   133  -134  5084    53    52   -51   -50   111  -112    86    87    96   -95

MP...DATA FOR EACH TYPE OF ZONE...
     1     5     3     2     2     2     0    22    12     6     2     2     2     2     2     2     2    10     6     3     2     2     2     0   121     3
     2     1     2     3   127    13     7     2     2     2     1     2     2     2

LENGTH OF NE =  157          NO. OF REA. NETS =  12          NO. OF ZONES  =  12
LENGTH OF MP =   33          NO. OF TUBES     =   9          NO. OF TYPES  =   5
LENGTH OF  P =  217          NETS MONITORED   =   9          NO. OF MIRRORS =   3

NETS MONITORED...
     1     2     3     6     4     7     5     8     9

FINITE ELEMENTS PER NET...
     1     5     5     6     6     6    12     9     3     1     1     0
) DA2171
/R
TIEMPO 00 04 00      4       0
```

APPENDIX C - Future development

New types of bodies can be incorporated in the program as the need arises; for example regular or irregular polygons of which the interior may or may not be considered as an enclosure. At present, the interior of a circular net is considered to be external to the system, but it would be useful to provide the option of including the interior of closed nets.

At present, the operation of division of logical expressions is limited to simple cases, which must be anticipated by the user. A complete reorganization of the data structure would be needed in order to permit division in the general case, but such a modification would make the program much more powerful.

Multiply connected enclosures are not permitted; if such an enclosure is specified, the results produced are erroneous and unpredictable. The program could be made more general in this respect, but would possibly need some "guidance" from the user in the form of data to indicate where multiply connected zones are to be found.

This type of problem is well suited for resolution on an interactive graphics terminal. Not only would the data entry be made easier and safer, but the program execution could be controlled by the user in visual form, for example by watching the various enclosures high-lightd on the screen, one by one.

APPENDIX D - Data Sheets

Blank data sheets are included for the user's convenience. Copies are available from the computer centre of the "Centro Atômico Bariloche".

SPADE DATA PREPARATION FORM (SHEET 1)

HEADING:

CODES:

CODE C	VALUE V
-1	

MATERIAL PROPERTIES:

	EMISSIVITY	INITIAL TEMP °C
1		
2		
⋮		
-1		

AUXILIARY POINTS:

	RELATIVE TO: P	Mx	Cx	My	Cy	REPETITION N
1						
2						
3						
4						
5						
⋮						
-1						

NOTE: PUNCH ALL DATA ENCLOSED IN BOXES.

SPADE DATA PREPARATION FORM (SHEET 2)

PROGRAM CONTROL:

MAX D	MIN D	MPAK	MNUM	PLOT

CIRCULAR NETS:

TYPE T	MATERIAL M	SIDES/FE S	CENTRE C	RADIUS R	ELEMENTS E	REPETITION N	EQUIVALENCE, Q
0					12		

LINE SEGMENTS:

TYPE T	MATERIAL M	SIDES S	END-POINTS A	END-POINTS B	GENER.% G	H*	EQUIVALENCE, Q
2							
-1							

*H = NODE No. FOR G ≠ 0
ARC CENTRE FOR G = 0

NETS TO BE MONITORED:

-1							

REFERENCES

1. S.Pissanetzky, "Numerical simulation of the transient temperature distribution inside a close-packed array of cylindrical tubes during heating and cooling under high vacuum". Nuclear Engineering and Design 56 (1980) 359-368.

2. O.C.Zienkiewics, "The finite element method in engineering science", McGraw-Hill, London (1971).

3. R.Siegel and J.R.Howell, "Thermal radiation heat transfer", McGraw-Hill, New York (1972).

4. G.M.Crippen, "A novel approach to calculation of conformation: distance geometry". Journal of Computational Physics 24 (1977) 26.