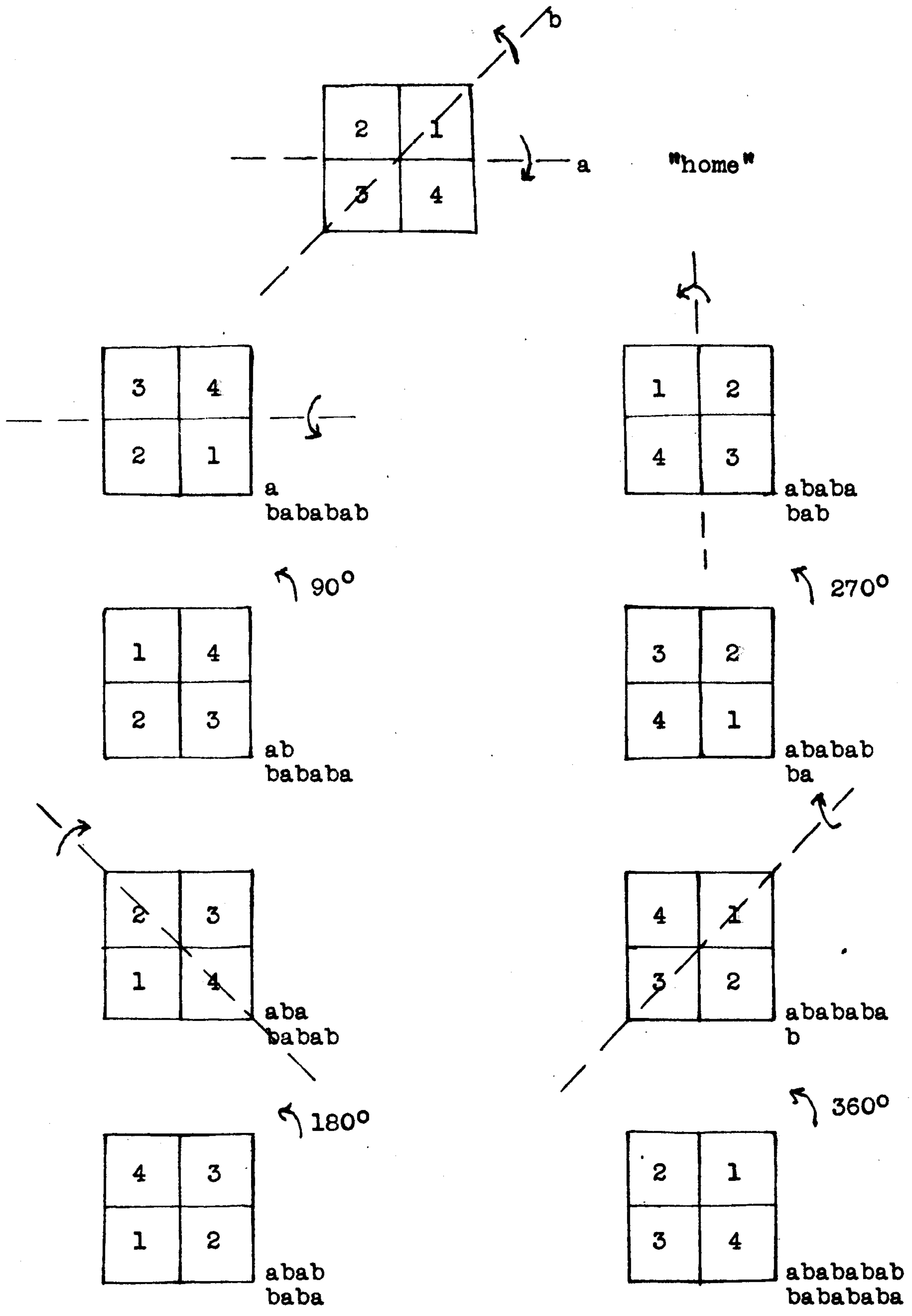
Appendix I: Symmetry

Symmetry In The Plane

Before attacking the problem of symmetry in the complete game, it is helpful to examine the symmetries in the two dimensional configuration. The first consideration is the number of ways that the square can be orientated so that the distance relationships of the corners remain unchanged, regardless of what may happen to the center. A simple way to approach this problem is to pick any particular corner and assume it is occupied by a particular point. This fixes the opposite corner and leaves two possible alternatives for assigning the two remaining corners. There are four possible points which could have occupied the corner originally, so there are four times two, or eight symmetrically equal configurations of the square itself. This could also have been seen by realizing that there are four equal positions which arise from rotations of 90, 180, and 270 degrees around the center point, two which arise from rotation around the "center" lines, and two which arise from rotation around the diagonals. Figure IV shows these eight configurations and a method of generating all eight by systematic application of only two symmetric operators, rotation around a "center" line and rotation around a diagonal.

Next consider the possibility of keeping the four corner points fixed and all ten line relationships the same, but changing the location of the remaining twelve points. Figure Vb shows a way in which the base position of Figure Va can be arranged so that the above restrictions are met. This symmetry function can now be applied to each of the

Figure IV: Eight Basic Square Symmetries



previous eight positions, resulting in sixteen symmetrically equal positions of the four by four square. These positions can be generated by the following series, where a and b are the operators of Figure IV and c is the new "constant corner" operator: acbcacbcacbcacbc.

An analysis of the sixteen squares shows that each falls into one of the following three classes:

Type	Straight Lines	Diagonals
Inside	2	1
Corner	2	1
Other	2	0

The fact that the corner squares and inside squares lie on the same number and type of lines indicates that a symmetry may also exist which maps the inside points into the corners and the corners into the inside. By simply assigning a corner point to an inside point, the remainder of the points can be filled in to form the configuration shown in Figure Vc. It would have been possible to fill in these points in several other ways, but all of these configurations would have been symmetrically equal to the one in Figure Vc by application of one of the symmetries previously discussed.

This new symmetry can be applied to each of the previous sixteen positions, resulting in a total of 32 symmetrically equal representations of the four by four array, and also showing that there are only two distinguishable types of points. The series of symmetrical operators which will produce these 32 positions can be obtained by inserting the new operator between each pair of operators in the above series.

Figure V: Other Planar Symmetries

00	01	02	03
10	11	12	13
20	21	22	23
30	31	32	33

a: Base Configuration

00	02	01	03
20	22	21	23
10	12	11	13
30	32	31	33

b: Constant Corner Symmetry

11	10	13	12
01	00	03	02
31	30	33	32
21	20	23	22

c: Inside-Outside Symmetry

Symmetry In The Full Array

The problem of symmetry in the full game can be attacked in the same manner used in finding the symmetries of the square, The first consideration is the total number of symmetries in the cube itself, apart from internal considerations. Again a corner is picked and the possible configurations of the corners which must be unidistant from that point are considered. There are three of these corners and therefore 3!, or six possible configurations. There were eight points which could have been chosen originally, so there are six times eight, or 48 symmetries of the cube itself. Table I lists these symmetries and how they can be generated by successive application of three basic operators. The values of X, Y, and Z are the binary coordinates of the corners.

A similar "constant corner" symmetry also exists in the three dimensional case. The eight corner points remain fixed and the inside center points are transformed. Since the cross-corner diagonals must be maintained, the only possible variation is a switch of the inner pairs of each diagonal. This results in changes of the outside non-corner points, and the entire mapping can be described by the operator $X_rY_rZ_r$, where the operator A_r exchanges the high and low order bits of the two bit binary number A. This operator can be applied to each of the previous 48 positions, resulting in a total of 96 symmetrically equivalent positions.

An "inside-outside" symmetry also exists in the three dimensional case, and is described by the operation $X_1Y_1Z_1$, where the operator A_1 inverts the low order bit of the two bit binary number A. Application of this operator to each

Table I: Symmetries of the Cube

a = Y X Z   b = X Y' Z   c = Z X' Y'

	X Y Z
a	Y X Z
b	Y X'Z
a	X'Y Z
b	X'Y'Z
a	Y'X'Z
b	Y'X Z
a	X Y'Z
c	Z X'Y
a	X'Z Y
b	X'Z'Y
a	Z'X'Y
b	Z'X Y
a	X Z'Y
b	X Z Y
a	Z X Y
c	Y Z'X'
a	Z'Y X'
b	Z'Y'X'
a	Y'Z'X'
b	Y'Z X'
a	Z Y'X'
b	Z Y X'
a	Y Z X'
c	X'Y'Z'
a	Y'X'Z'
b	Y'X Z'
a	X Y'Z'
b	X Y Z'
a	Y X Z'
b	Y X'Z'
a	X'Y Z'
c	Z'X Y'
a	X Z'Y'
b	X Z Y'
a	Z X Y'
b	Z X'Y'
a	X'Z Y'
b	X'Z'Y'
a	Z'X'Y'
c	Y'Z X
a	Z Y'X
b	Z Y X
a	Y Z X
b	Y Z'X
a	Z'Y X
b	Z'Y'X
a	Y'Z'X
c	X Y Z

of the previous 96 positions yields 192 symmetrically equivalent positions.

This inside-outside operation is also important since it maps inside center points into corners and outside center points into interior edge points. This reduces the number of distinguishable points to two, as in the planar case.

Appendix II: Calculation of the Total Number of Positions

To calculate an extremely rough upper bound on the total number of positions of the board, it can be assumed that each square can have one of three possible conditions, occupied by the first player, occupied by the second player, or unoccupied. This results in $(3)^{64}$ total positions, or about 3.5×10^{30}.

This formulation completely ignores one of the most obvious restrictions of the game, however, that the total number of squares occupied by the first player must be equal to or one greater than the total number of squares occupied by the second player, depending on who is to move. Since only an approximation is required, it will be assumed that the total number of positions is the same for both situations, and only the total number of positions when the first player is to move will be calculated. This figure can be obtained by calculating the number of configurations when each player has no pieces, one piece, etc., and summing these results. This can be expressed mathematically by the expression

$$\sum_{n=0}^{32} \left(\frac{64!}{(64-n)!\,n!}\right)\left(\frac{(64-n)!}{(64-2n)!\,n!}\right)$$

The result of this calculation will be a sum of the form

$$1+\frac{64\times63}{1\times1}+\frac{64\times63\times62\times61}{2\times1\times2\times1}+\cdots+\frac{64!}{32!\times32!}$$

which can be factored as

$$\left(1+\frac{64\times63}{1\times1}\left(1+\frac{62\times61}{2\times2}\left(1+\frac{60\times59}{3\times3}\left(\cdots+\frac{2\times1}{32\times32}\right)\cdots\right)\right)\right)$$

Evaluation of this expression on the TX-0 gave a result of 0.10 101 000 110 100 110 x 2^{1428}, or about 2.09 x 10^{29}. This figure is exactly the number of ways in which the board can be arranged when the first player is to move, but contains a great many configurations which are of no interest because the game has already been won, possibly several times by one or both players.

An upper bound on the total number of positions which contain four in a row can be calculated in the same manner. Four of the squares of one player are constricted to lie on a line, and the number of ways the remainder of the pieces can be located are calculated. This sum is given by the expression

$$\sum_{n=4}^{32} \left(\frac{60!}{(60-n)!\ n!}\right)\left(\frac{(60-n)!}{(60-n-n+4)!\ (n-4)!}\right)$$

A similar evaluation gave a result of 2 x 10^{27}. This figure must now be multiplied by 76, since the four in a row could have been on any one of the 76 lines, resulting in an upper bound of 1.52 x 10^{29}. This is an upper bound because the restriction is not made that no new four in a row be formed with the remaining pieces, and each position will be counted as many times as it has four in a row.

A lower bound on the total number of "active", or non-won, positions can now be calculated by subtracting the upper bound on the number of positions which contain at least one four in a row from the total number of positions. This results in a figure of 5 x 10^{28}. If these positions were actually going to be stored in the machine, a great deal of space could be saved by storing symmetrically equal positions only once. A lower bound on the number of positions which would then need to be stored is given by dividing the lower bound on the number of positions, 5×10^{28}, by the number of symmetries, 192, which yields about 2.5×10^{26} different positions. This figure is certainly a lower bound since there are not 191 positions which are symmetrically equal to a given position, for instance the position with no pieces for either player is symmetrically equal only to itself.

This figure can now be multiplied by two, since it contains only the positions when the first player is to move, giving a final total of 5×10^{26}. If one bit of information was to be stored about each of these positions, with a packing density of 200 bits per cubic inch, the memory required would be approximately the size of the earth.

An attempt was made to further reduce this figure by removing all positions which contained a line where a three to nothing situation existed, since the choice of a move in these positions is trivial. This upper bound was calculated in the same manner as that of the four in a row case and contained similar duplication. Because of the high amount of duplication the figure came out about 4×10^{26}, resulting in a negative number of active positions, and was therefore discarded.

It is also interesting to estimate the number of different possible games. An upper bound can be obtained as 64!, or about 10^{89}, but this is clearly much too large. A lower bound can be obtained by assuming that each possible game goes through 64 different positions and that no one position is encountered in any two games. Then by dividing the lower bound of the number of active positions by 64, a lower bound of 5×10^{24} is obtained.

Appendix III: The Mathematical Description Of The Game

The 64 squares of the game are coded by giving their coordinates on a three axis cartesian system. The x and y coordinates specify the square's location in the horizontal plane and the z coordinate specifies the horizontal plane in which the square is located. Each coordinate is expressed with a pair of binary digits and the three pairs are combined into a six bit binary number in the following manner, $z_2z_1y_2y_1x_2x_1$. For input-output purposes, these six bits are represented as two octal digits, with the high order octal digit being composed of $z_2z_1y_2$ and the low order octal digit being composed of $y_1x_2x_1$.

There are 76 different four square lines in the game, which can be broken down into the following types:

Type	Mathematical Description	English Description	Number
A	Orthogonal to two axis	Straight Lines	48
B	Orthogonal to one axis	Regular Diagonals	24
C	Orthogonal to no axis	Cross Corner Diagonals	4

Since a cartesian system is being used to number the squares, all lines can be described by a point on the line and the slope of the line. In particular, when the point is chosen to be the square with the lowest octal value and the slope is expressed with the same xyz convention as was used to number the squares, the other three points on the line can be found by successively adding the slope to the base point. Therefore all lines are described by four octal digits, the first two specifing the base point and the second two the slope. These descriptions of the lines are given in Table II, where the subscript on the type A

Table II: Description of Lines

Line Number	Type	Base Point	Increment	Diagonal #
00	A_x	00	01	
01		04		
02		10		
03		14		
04		20		
05		24		
06		30		
07		34		
10		40		
11		44		
12		50		
13		54		
14		60		
15		64		
16		70		
17		74		
20	A_y	00	04	
21		01		
22		02		
23		03		
24		20		
25		21		
26		22		
27		23		
30		40		
31		41		
32		42		
33		43		
34		60		
35		61		
36		62		
37		63		
40	A_z	00	20	
41		01		
42		02		
43		03		
44		04		
45		05		
46		06		
47		07		
50		10		
51		11		
52		12		
53		13		
54		14		
55		15		
56		16		
57		17		

Table II: (con.)

Line Number	Type	Base Point	Increment	Diagonal #
60	B_z	00	05	00
61		20		01
62		40		02
63		60		03
64		03	03	04
65		23		05
66		43		06
67		63		07
70	B_y	00	24	10
71		01		11
72		02		12
73		03		13
74		14	14	14
75		15		15
76		16		16
77		17		17
100	B_x	00	21	20
101		04		21
102		10		22
103		14		23
104		03	17	24
105		07		25
106		13		26
107		17		27
110	C	00	25	00
111		03	23	01
112		14	15	02
113		17	13	03

lines denotes the axis to which the line is parallel and the subscript on the type B line denotes the axis to which the line is orthogonal.

The present state of the game is stored in the machine by assigning one word to each of the 100_8 squares and 114_8 lines. In the 100_8 words assigned to the squares a +3 indicates that the square is unoccupied, a -3 that it is occupied by the machine, and a -2 that it is occupied by the player. This coding allows the "state" of any line to be uniquely determined by a simple arithmetic summation of the four points on that line. These sums are kept in the 114_8 registers assigned to the lines. The possible values of this summation and the corresponding situations are found in Table III.

In order to make lookahead routines run as fast as possible, it is highly desirable to be able to update only the line sums affected by a particular move, rather than recalculate the entire table. It is therefore necessary to have a fast way of generating all lines which pass through a given square. Because of the way in which the lines have been numbered, it is possible to generate the numbers of the three type A lines which pass through any square directly from that square's numerical value. The number of the A_z line is $y_2y_1x_2x_1+40_8$, the A_y line $z_2z_1x_2x_1+20_8$, and the A_x line $z_2z_1y_2y_1$. These are simple calculations to perform and leave only the problem of finding the diagonals, or the type B and C lines.

Although this information could also be generated directly from the points, it is faster to store a table in the machine which lists the various B and C types for each point.

Table III: Line Sum Values

Sum	Machine	Player	None	Situation
-12	4	0	0	Machine Won
-11	3	1	0	Dead
-10	2	2	0	Dead
-9	1	3	0	Dead
-8	0	4	0	Machine Lost
-7				Impossible
-6	3	0	1	Machine Can Win In One Move
-5	2	1	1	Dead
-4	1	2	1	Dead
-3	0	3	1	Player Can Win In One Move
-2				Impossible
-1				Impossible
-0	2	0	2	Machine Can Force Player
1	1	1	2	Dead
2	0	2	2	Player Can Force Machine
3				Impossible
4				Impossible
5				Impossible
6	1	0	3	Machine Has Possibilities
7	0	1	3	Player Has Possibilities
8				Impossible
9				Impossible
10				Impossible
11				Impossible
12	0	0	4	Unoccupied

Since there are 24 type B lines, five bits can be used to identify them, and two bits to identify the four type C lines. In order to obtain the line number, 60_8 is added to the type B number and 110_8 to the type C number.

In the machine a six digit octal number is stored for each square. As was previously shown, there are only two types of points, which are either on one B and no C or three B's and one C. If the point is of the former type, the four high order octal digits are zero and the B number is contained in the low order two octal digits. If it is of the latter type, the high order octal digit is four plus the C line number and the low order five octal digits are the three B line numbers combined $B_zB_yB_x$. These numbers are given in Table IV.

Table IV: Point To Line Constants

Point	Number	C	B_z	B_y	B_x
00	400420	0	00	10	20
01	11			11	
02	12			12	
03	510564	1	04	13	24
04	21				21
05	00		00		
06	04		04		
07	25				25
10	22				22
11	04		04		
12	00		00		
13	26				26
14	610623	2	04	14	23
15	15			15	
16	16			16	
17	700767	3	00	17	27
20	01		01		
21	20				20
22	24				24
23	05		05		
24	10			10	
25	402461	0	01	11	21
26	512525	1	05	12	25
27	13			13	
30	14			14	
31	612662	2	05	15	22
32	702726	3	01	16	26
33	17			17	
34	05		05		
35	23				23
36	27				27
37	01		01		
40	02		02		
41	24				24
42	20				20
43	06		06		
44	14			14	
45	704665	3	02	15	25
46	614721	2	06	16	21
47	17			17	

Table IV: (con.)

Point	Number	C	B_z	B_y	B_x
50	10			10	
51	514466	1	06	11	26
52	404522	0	02	12	22
53	13			13	
54	06		06		
55	27				27
56	23				23
57	02		02		
60	706624	3	03	14	24
61	15			15	
62	16			16	
63	616760	2	07	17	20
64	25				25
65	03		03		
66	07		07		
67	21				21
70	26				26
71	07		07		
72	03		03		
73	22				22
74	516427	1	07	10	27
75	11			11	
76	12			12	
77	406563	0	03	13	23

Appendix IV: Some Rules of Forcing Wins

One of the most important known strategical rules about Qubic concerns planar arrangements which will allow a forced win in that plane. These situations are taken to be independent of the state of the rest of the game, which may or may not be important, depending on the state of the lines which intersect squares on which the opponent is forced. If the opponent generates a three in a row on a line outside the plane, then that line must be blocked and the simple planar situation no longer exists, but this does not happen frequently.

Figure VI shows all possible arrangements of three squares in a plane, after symmetry reductions. The configurations are classified as three in the core squares, a-d; two in the core squares, e-n; one in the core squares, o-x; and none in the core squares, y-cc. Fourteen of these 29 configurations result in a forced win and those wins are indicated in red. Of these fourteen wins, three are trivial configurations where a three in a row existed at the start, while of the fifteen non-wins, five contain no two in a row and are therefore trivially non-forcing. Of the remaining 21 "interesting" configurations, eleven are wins and ten are not.

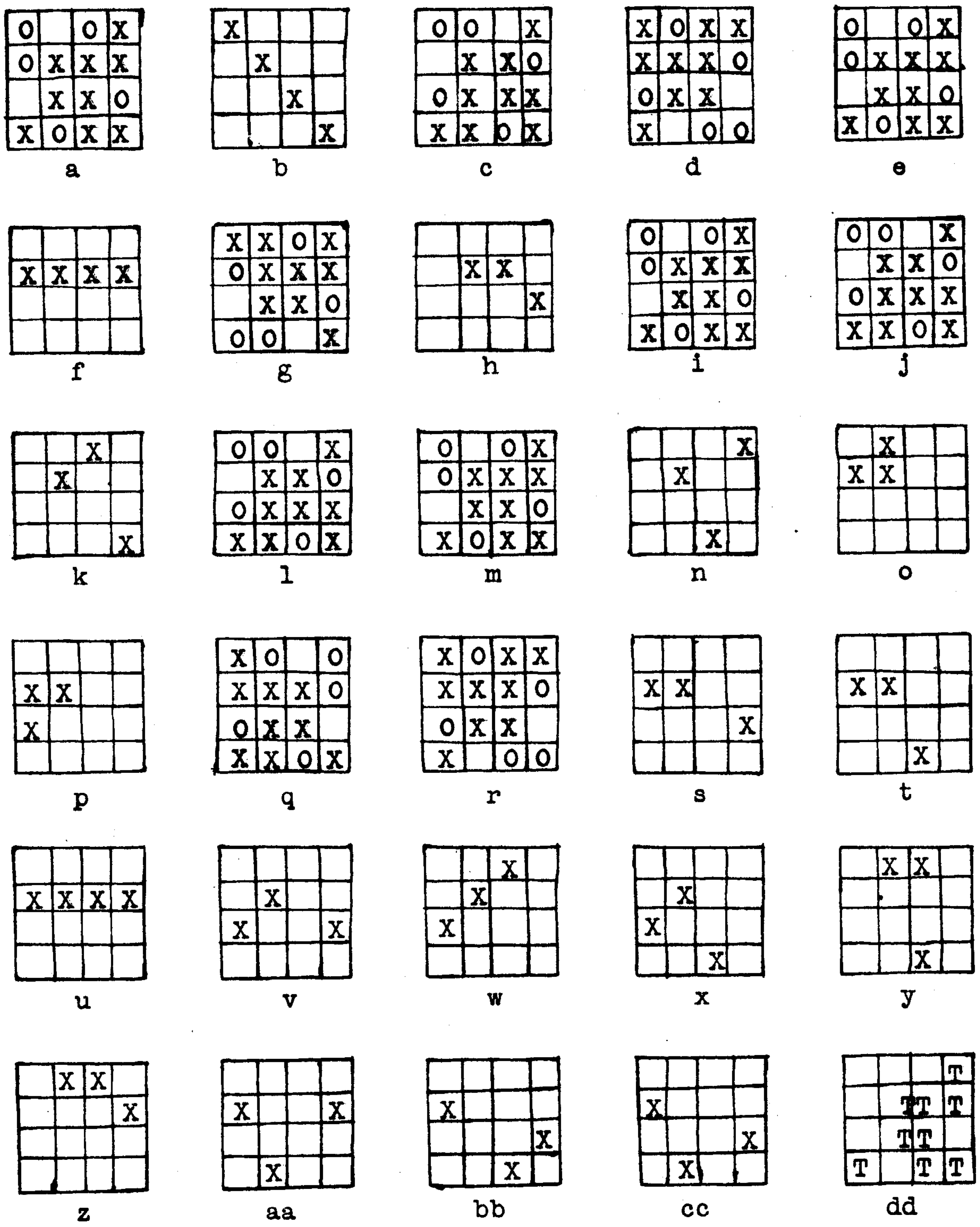
Configuration dd shows a normalized set of nine squares from which the three must come if the configuration is to be forcing. All sets of this set are not necessarily forcing, for some do not even contain a two in a row. Of the 15 possible configurations that can come from this set, eleven are forcing.

A more general rule can be stated for the entire game which covers many forcing situations directly and which will apply to all forcing sequences at some time during the process of carrying them out. Given any two lines which contain two in a row, if a sequence of lines exists which contain one in a row and intersect each other on empty squares, and two of these lines intersect the two lines with two in a row on empty squares, then a forced win can be executed, if the forces do not establish a force for the opponent, as discussed before. The procedure for following this force is quite simple, first force on one of the two in a row lines and choose the square which is on the intersection with the one in a row line. This establishes the basic position again but the number of one in a row lines has been reduced by one. The repeated application of this process will eliminate all the one in a row lines and the final move will develop three in a row on two lines for the win.

Considerable ingenuity can be used in order to develop sequences which curve back on themselves in such a way as to minimize the ratio of number of pieces needed to number of lines in the sequence. It is also possible to fold the lines back in such a manner that only one two in a row is needed to start with. Consider the configuration of pieces 12, 32, 61, 64, 73, and 76. The only existing two in a row is on the vertical line through square 12, where forcing moves can be made at squares 52 and 72. By choosing 72 as the forcing move, two new two in a rows are developed in the top plane and the previous conditions are met. Consecutive forces at squares 70, 60, and 62 will produce a win at either square 73 or square 76.

Although these rules are quite interesting and useful to the human player, they were not particularly suited for use by the machine and were not included in the program.

Figure VI: Planar Forces


a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v
w
x
y
z
aa
bb
cc
dd