6

AUTOMATIC DESCRIPTION AND RECOGNITION OF BOARD PATTERNS IN GO-MOKU

A. M. MURRAY and E. W. ELCOCK
UNIVERSITY OF ABERDEEN

INTRODUCTION

A series of computer programs have been written to play the board game Go-Moku. Go-Moku is played on a 19 × 19 square mesh. Player b(w) has a supply of black (white) pieces. The players take it in turns to play a piece on a mesh point. The winner is the first player to complete a *5-pattern*, that is, to make up a (horizontal, vertical or diagonal) line of five and only five adjacent pieces of his colour.

The programs carry out a backtrack analysis on games which have been lost to a human opponent. These analyses automatically generate a list of descriptions of subgoals, each hopefully describing the essential structure of a board pattern from which completion of a 5-pattern is inevitable. The playing capability of such a program depends critically on the power of the abstract descriptions and on the machinery for recognising realisations of described subgoals.

A distinct phase of the work was completed for the Machine Intelligence Workshop held under the auspices of the Experimental Programming Unit in Edinburgh in 1965. This phase included complete and working programs which allowed investigation and evaluation of a number of simple backtrack analysis routines. The learning programs resulting from this first exploratory phase automatically become competent players after a small number of games played against a human opponent. They reach a point, however, still far short of expert play, beyond which they cannot develop further. Their inadequacies did, however, highlight the problems that needed to be solved for more continuing self-improvement to be possible.

The paper by Elcock and Murray (1967) describing this work was published in the proceedings of the Workshop and provides useful background to the present paper.

The last year has been spent on exploring solutions of some of the problems highlighted by the earlier programs. A more sophisticated analysis routine which works in the context of a more powerful and general way of describing board patterns has been designed and implemented. The new programs are capable of analysing winning positions and, from the analysis, synthesising descriptions of subgoals of arbitrary complexity. An interesting method has been developed for automatically generating, from a description of a subgoal obtained by the analysis/synthesis process, a segment of control instructions which, when interpreted, direct the processing of the board required to recognise realisations of the described subgoal.

The present paper reports on the part of this work that is concerned with description and recognition.

DESCRIPTION OF BOARD PATTERNS

This section gives an informal discussion of the aspects of board patterns described and explains why the particular description scheme chosen is appropriate both to play of the game, and particularly to automatic acquisition of descriptions by program analysis of played games.

First, a few general remarks. A description says something about a more or less local pattern of played pieces on the board. It is *not* a representation: it does *not* attempt to say all that can be said about a particular local pattern in the sense that the particular pattern could be reconstructed from the description. The things a description does say express certain constraints (conditions) that a local pattern must satisfy in order to meet the description. The particular kinds of constraint that can be expressed are, hopefully, those that allow any particular board situation which contains a winning move to be described in a way which captures constructively the essential content, and only the essential content, which makes the win inevitable. Such a description, once acquired, should then be useable in game play to recognise any one of a class of board positions which in some local region meet the constraints of the description and which contain a recognisable winning move.

A description makes no reference to colour and can describe either a black object or a white object. In the text and diagrams that follow, white objects will be used throughout: corresponding black objects are obtained simply by replacing black by white and white by black.

Motivation of the descriptive scheme

The next few paragraphs try to expose, by means of examples, the motivation underlying the descriptive scheme implemented. In all diagrams only the relevant region of a board is given.

Example 1. In any of the board situations of Fig. 1 white play at node 1

completes a 5-pattern (and wins immediately). The essential content of a board situation which makes immediate completion of a 5-pattern possible is, trivially:

> there exists a node which is a constituent of a possible 5-pattern, of which 4 pieces have already been played, on some one line through the node

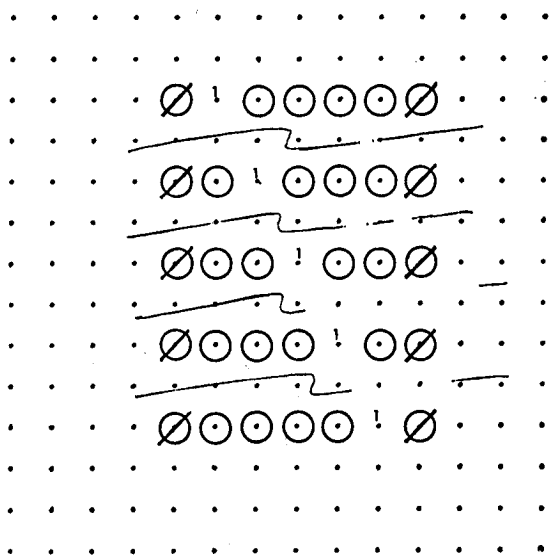


FIG. 1. Ø not a white piece.

Example 2. In any of the board positions of Fig. 2 white play at node 1 has the consequence that a 5-pattern can be completed at the next white move whatever the intervening black move: if black plays node 1′ (1″) then white plays 1″ (1′).

The essential content of all these board situations is:

> there exists a node which is a constituent of two possible 5-patterns, in each of which three pieces have already been played, on some one line through the node.

Attention is drawn to the fact that, as well as the node referred to explicitly, the object (as described by the text after 'there exists') contains, as it should, two implicit nodes (the nodes 1′ and 1″ of the representations of the object in Fig. 2).

It should be noted that, if it is the opponent to play, he can destroy a realisation of the object by play at any of its explicit or implicit nodes.

Consider the board situation:

Ø..:000:.Ø

The position contains two realisations of the object described above: one with α as the explicit node and one with β as the explicit node. These two

realisations have nodes in common (β is an implicit node for the realisation of which α is the explicit node . . .). With black to play he can destroy both realisations by playing at a common node.

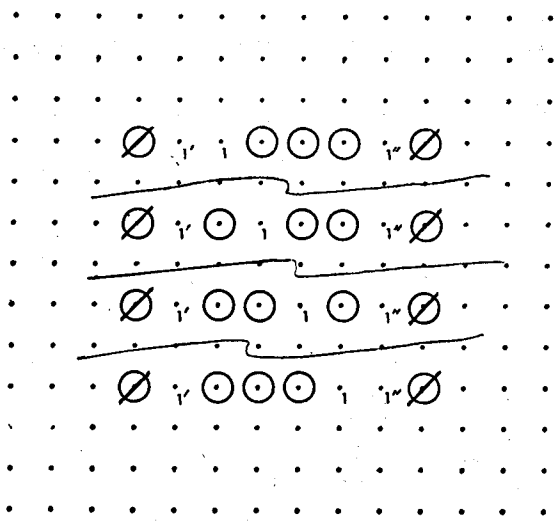


FIG. 2

Example 3. Fig. 3 shows a board position in which, after play at node 1, there are again two realisations of the object described in Example 2. This time, however, they do not share a common node; black cannot destroy both and white has a sequence of moves which inevitably lead to completion of a 5-pattern.

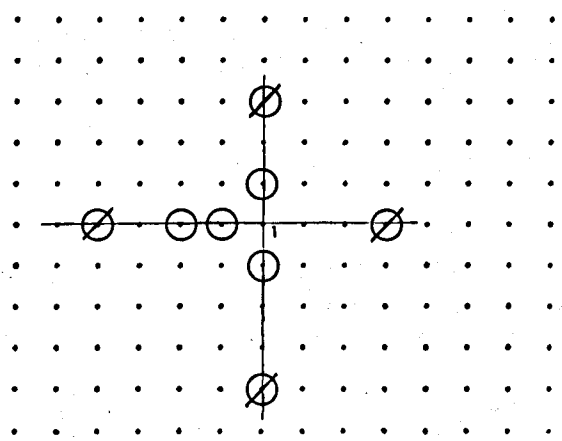


FIG. 3

The essential content of this (and similar) board positions is:

there exists a node which is a constituent of two possible 5-patterns, with two pieces played, on each of two lines through the node.

The description implies six other nodes; three on each line through the explicit node.

Play at the explicit node creates a set of realisations of the object described in Example 2 with no node common to all of them and which consequently cannot all be destroyed by the opponent's next move.

Example 4. Fig. 4 shows a board position in which, after play at node 1, there are two realisations of the object described in Example 1. A 5-pattern can always be completed in two player's moves.

The essential content of this class of board positions is:

> there exists a node which is a constituent of a possible 5-pattern, with three pieces played, on each of two lines through the node.

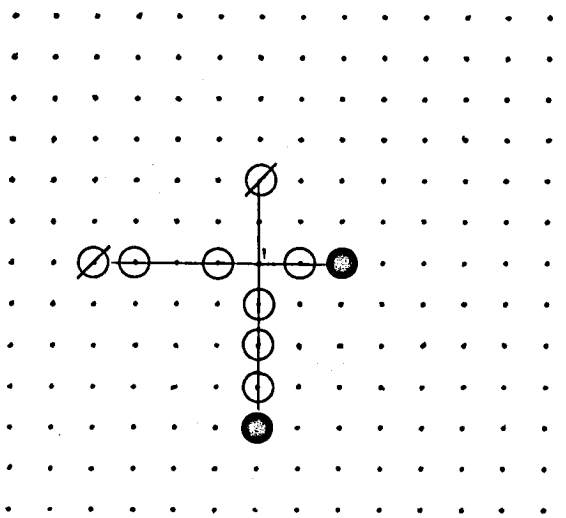


FIG. 4

Examples 5 and 6. So far the descriptions have been very simple: each has had just one explicit node and, in a realisation of the description, play at the explicit node initiates a sequence of moves leading inevitably to the creation of a 5-pattern.

Consider, however, the board positions of Figs. 5 and 6.

In the board position of Fig. 5, after play at node 1 a realisation of the objects described in Examples 2 and 3 is created at nodes 1′ and 2 respectively. Similarly, play at node 2 creates realisations of the same objects at nodes 1 and 2′ respectively.

The description (in words) of the essential content of this kind of board situation is rather lengthy:

> there exists a node (1) which is a constituent of two possible 5-patterns on each of two lines through the node: on one of the lines through the node the patterns have two pieces played: on the other line the patterns

have one piece played and a common implicit node (2) of these patterns is itself a constituent of two possible 5-patterns, with two pieces played, on some other line through it.

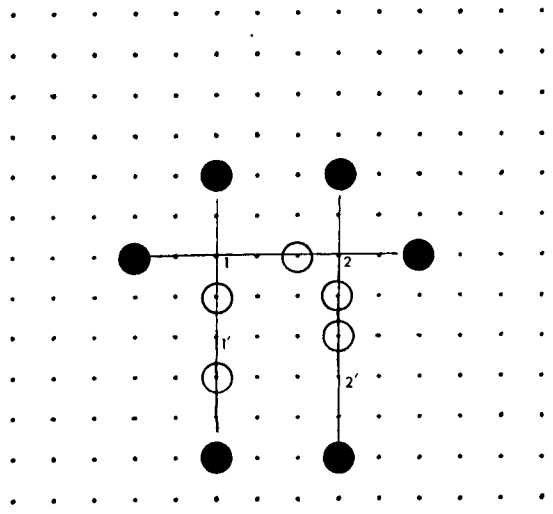


FIG. 5

The description makes explicit reference to two nodes. Nodes such as 1′ and 2′ of the realisation of diagram 5 are implied by the description.

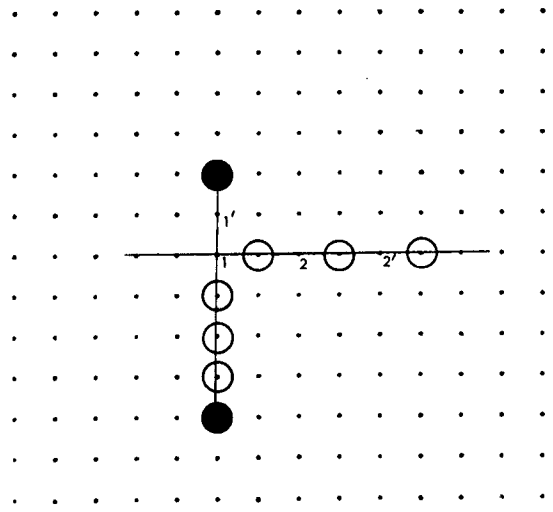


FIG. 6

In a realisation of this description the labelling of the explicit nodes 1 and 2 is arbitrary: the realisation has a certain symmetry. Play at either of the

explicit nodes of a realisation initiates a sequence of moves which leads to creation of a 5-pattern in at most four player's moves.

Contrast this with the board situation in Fig. 6.

After play at node 2, a realisation of the objects described in Examples 1 and 4 is created at nodes 2′ and 1 respectively.

The essential content of this kind of board situation is:

> there exists a node (1) which is a constituent of a possible 5-pattern on each of two lines through the node: on one of the lines the 5-pattern has three pieces played: on the other line the 5-pattern has two pieces played and an implicit node (2) of this pattern is itself a constituent of a possible 5-pattern on the same line, but with three pieces played.

The description again makes explicit reference to two nodes. This time, however, the labelling of the nodes 1 and 2 in a realisation is not arbitrary and, in fact, play at node 1 does not initiate a sequence of moves leading inevitably to the completion of a 5-pattern.

In the descriptive scheme implemented it is possible to add 'comments' to explicit nodes of a description. The 'comments' take the form of a statement of the expected number of moves required to complete a 5-pattern from play at a realisation of the object. As will be described later, these comments are made by program during the automatic acquisition of the description.

Automatic acquisition of descriptions

All the descriptions given in these examples imply that in a realisation of the description there exists a continuation which leads inevitably to the creation of a 5-pattern by the player. It is just the existence of this continuation that the description is designed to capture.

The described objects have this property because play at an appropriate node of a realisation creates a set of realisations of describable simpler objects, each of which has the property, and which do not share a common (implicit or explicit) node.

Descriptions with this property are called descriptions of 'subgoals'.

Most important is the implication of this property of the descriptive scheme for easy automatic acquisition of descriptions of subgoals by program.

Assume that the program can recognise realisations of described subgoals. Consider a stage in the program's game playing where the subgoals of Examples 2 and 3 have been described. Suppose the current state of the board in a game being played is that shown in Fig. 7. It is the opponent to play. At its last move the program did not recognise any realisations of described subgoals.

The opponent plays at node 1. The program now recognises the following realisations of known subgoal descriptions:

(i) a realisation of Example 2 at a with implicit nodes a' and b;
(ii) a realisation of Example 2 at b with implicit nodes a and b';
(iii) a realisation of Example 3 at c with implicit nodes $c', c'', c''', d', d'', d'''$.

An analysis of the situation might go as follows:

The realisations (i), (ii) and (iii) do not share a common node and therefore the opponent can inevitably create a 5-pattern.

A minimal set of realisations with this property is (i) and (iii). The piece at 1 must be a common played piece of each member of this set.

From this analysis it is reasonably straightforward to synthesise a description of the subgoal of which there was a realisation one move back in the game.

It is necessary to identify, for each subgoal realisation of the minimal set, the relevant lines for which the piece at 1 is a played piece. It is then necessary to (i) downgrade (by effectively unplaying this piece) the components of the description that refer to these lines, and (ii) make node 1 an explicit node of

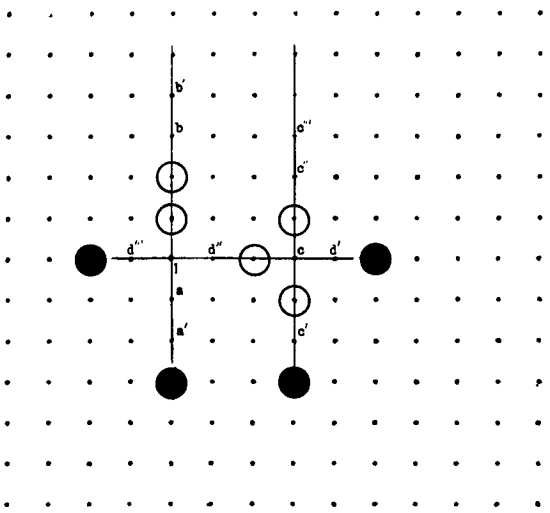


FIG. 7

these downgraded descriptions. A new description can now be synthesised from these downgraded descriptions by taking the union of them through the common node 1.

This description will describe a subgoal for which there is the realisation in the particular board situation from which the description was abstracted. Once the description is acquired, any of the whole set of its realisations can be recognised in future games.

It is not difficult to make this synthesis procedure precise and to automate it so that the program can, from game play such as the above, acquire descriptions of new subgoals. By the very nature of the process, this acquisition of descriptions of subgoals gives the program an increasing capability for directed play in board situations further and further away from a win.

RECOGNITION OF BOARD PATTERNS

The end result of being able to acquire descriptions of subgoals is to be able to recognise realisations of them in game play. This section discusses briefly

how this is done. Since one of the main interests here is the program implementation of the recognition process, it is necessary to introduce a notation for descriptions which is closer to the program level than the segments of English text given in the examples in 'motivation of the descriptive scheme'.

A formal notation for descriptions

As we have seen, a description specifies a number of explicit nodes and certain linear 5-pattern relationships between them. The nodes are labelled 1, 2, 3 ... A description consists of a 'node list' followed by a 'pattern list'.

The node list has an element for each explicit node (the 'parent' node of the element). The elements are listed in the serial order of the labels of their parent nodes. Thus

$$(1, 2, 3, 4-1, 3/4-1, 2/4-1, 2/3)$$

is the node list for a subgoal with 4 nodes, the elements of the node list being separated by '—'.

Each element on the node list is a 'line list' with up to four elements each referring to a different line through the parent node, elements of the line list being separated by ','. An element consists of the labels of those and only those explicit nodes which, in common with the parent node, contribute to a possible 5-pattern or patterns, specified in the pattern list, on the line. The label of the parent node itself is suppressed unless there would otherwise be a null entry.

Thus, the first element of the node list above states that node 1 is a node of specified (in the pattern list) patterns on four lines through it. On one line no other nodes are referenced. Nodes 2, 3 and 4 are referenced on one each of the other three specified lines. The second element states that node 2 is a node of specified patterns on two lines through it. On one line node 1 is referenced and on the other line nodes 3 and 4 are referenced. The third element, etc.

It should be emphasised that the node list specifies the line pattern intersections completely, i.e., none of the specified line patterns intersect in a node of any other specified pattern other than at the stated explicit nodes.

The pattern list has the same structure as the node list. There is an entry for each line element of the node list. The entry specifies the possible 5-pattern or patterns, of which the parent node is a constituent, that must exist on the line. The patterns are specified by an integer $n (0 \leqslant n \leqslant 5)$ equal to the number of pieces in the pattern after play at the parent node. If the node must be a constituent of two patterns with the given n, then this is indicated by dashing the value of n (e.g. 4').

Consider the complete description:

$$(1, 2, 3, 4-1, 3/4-1, 2/4-1, 2/3)$$
$$3', 2', 2', 2'-2', 1'-2', 1'-2', 1'$$

Node 1 must be a constituent of two possible 5-patterns, in each of which two pieces have been played, on a line through it which references no other nodes. It must be a constituent of a 2' on each of the three other lines through it

passing through nodes 2, 3 and 4 respectively. Node 2 must be a constituent of a 2′ on the line through it which passes through node 1, and a constituent of a 1′ on another line through it which passes through nodes 3 and 4, etc. A realisation of the described object is given in Fig. 8.

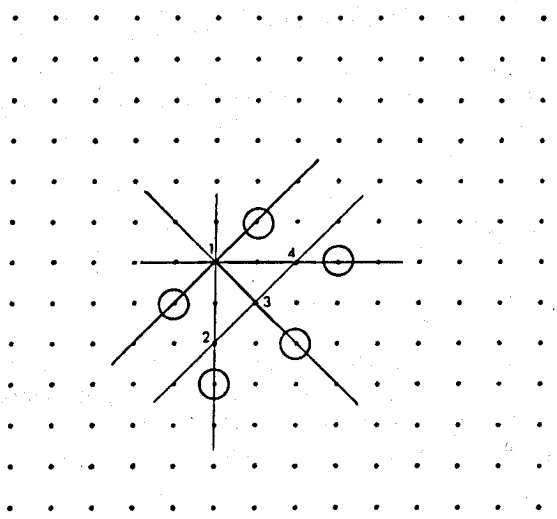


FIG. 8

Finally, comments may be added. As already mentioned (p. 81), the only comment for which provision is made in the present implementation is the prefixing of an element of the pattern list by a number equal to the expected number of player's moves needed to complete a 5-pattern in a realisation of the subgoal. Thus, the comment '6:' in

(1, 2, 3, 4−1, 3/4−1, 2/4−1, 2/3)
6: 3′, 2′, 2′, 2′−2′, 1′−2′, 1′−2′, 1′

for the subgoal of diagram 8.

A complex subgoal can be described in different equivalent ways corresponding to different labelling of nodes, etc. An arbitrary set of rules is used to ensure that descriptions of essentially the same object will always be given in a unique 'canonical' form.

In canonical form, the descriptions of the subgoals of the examples in 'Motivation of the descriptive scheme' are (note that subgoals having the same node list are grouped together):

(1)	
1:5	diagram 1
2:4′	diagram 2
(1, 1)	
2:4, 4	diagram 4

3:3′, 3′	diagram 3
.	
(1, 2−1)	
4, 3−3:4	diagram 6
.	
(1, 2−1, 2)	
4:3′, 2′−4:2′, 3′	diagram 5

The recognition processor

What has been given above is essentially an external (written) notation for descriptions. The program uses an equivalent representation in terms of a list structure. The written notation, or its equivalent list representation, exposes the structure of the subgoal and is particularly appropriate for the manipulations involved in the synthesis of new descriptions. It does not, however, make clear in any operational sense the sequence of actions that must be gone through to recognise a realisation of the described object in any particular board situation.

Recognition is facilitated by the use of a quite different representation of the description called the 'control stream' representation. Each control stream is made up of a sequence of control elements. Each control element in effect specifies a board processing instruction. The complete sequence of elements specifies the total processing that has to be gone through to recognise a realisation of the described object or, more precisely, to determine whether or not a local region of the board contains a realisation of the described object with a particular board node identified with node 1 of the object.

In what follows an example of a control stream is discussed informally and without going into too much detail. The intention is simply to bring out the difference between the synthetic and operational descriptions of subgoals and to present the view that the data structure which is the control stream can be thought of as a set of instructions in a board processing language.

Fig. 9 shows again a realisation of the subgoal of Fig. 8 and gives the written description together with the equivalent control stream. The format of the control stream is a feature of a particular machine implementation and is irrelevant to the present discussion.

In processing a particular board situation for realisations of the subgoal a particular vacant mesh point on the board is assigned the label "1" (i.e., is associated with node 1 of the description), and the control scheme is scanned by an interpreter.

An 'A' element has the structure:

(A code; pattern list element).

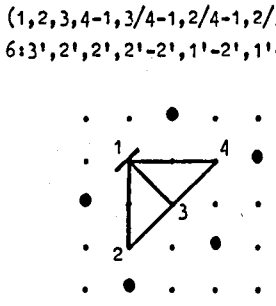
The first control stream element of the example of Fig. 9 is interpreted as 'check that mesh point "1" is a constituent of possible patterns meeting the specifications 3′, 2′, 2′ and 2′ on different lines through the mesh point: if successful, continue the scan'.

A 'B' element has the structure:

(B code; line pattern specification; node label; line label).

The second element of the example is interpreted as 'look for a line through the mesh point "1" for which the mesh point is a constituent of a pattern which meets the specification 3′; if successful, call this line "line 1 of node 1" and continue the scan'.

The third element of the example is interpreted as 'look for a line through the mesh point "1", not currently called "line 1 of node 1", for which the mesh point "1" is a constituent of a 2′: if successful, call this line "line 2 of node 1" and continue the scan'.



A	3'	2'	2'	2'				
B					3'			1,1
B					2'			1,2
C	2'	1'			2'	2		1,2
B					2'			1,3
C	2'	1'			2'	3		1,3
D				1'	1'	3,2		2,2
B					2'			1,4
C	2'	1'			2'	4		1,4
D				1'		4,2		2,2
D						4,1		3,2
E					3'			1,1
E	2				3'			1,2
E	3				3'			1,3
E	4				3'			1,4
E	4	3			3'			2,2

FIG. 9

A 'C' element has the structure:

(C code; pattern list element; line pattern specification; node label; node label; line label).

The fourth element of the example is interpreted as 'look for a vacant mesh point on line 2 of node 1, lying within a possible 5-pattern of which mesh point 1 is a constituent, which is a constituent of possible patterns meeting the specifications 2′ and 1′ on different lines through the new mesh point and which in particular meets the specification 2′ along line 2 of node 1: if successful, assign the label "2" to the mesh point and continue the scan'.

D-words are concerned with checking that already allocated nodes which are required to lie on a line are in fact on a line, and that the pattern specifications in the direction of this line are met.

E-words are concerned with finally checking that, for each line, the allocated nodes are in fact nodes of the same possible 5-pattern (the C-word allocations do not ensure this); checking the specification of this pattern, and listing implicit nodes (defensive moves). If the action of the last E-word is successful then a realisation of the subgoal has been recognised.

If at any stage it is impossible to continue the forward scan either because it is impossible to make the required allocation of a node or line, or after the last E-word to find other realisations with the same mesh point 1, the control

stream is backscanned by the interpreter to the first significant new allocation and the forward scan restarted.

CONTEXT AND SUMMARY

So far nothing has been said about how the control stream representation of subgoals is generated. In fact, there is a component of the total program which can transform the 'written' description, generated by the synthesis procedure, into the equivalent control stream.

With this in mind, Fig. 10 gives a schematic representation of the total program that has been written and of which some aspects of the pattern

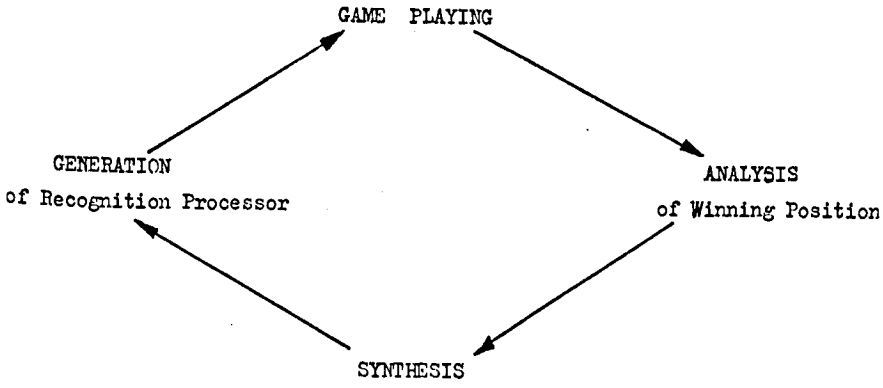


FIG. 10

description and recognition components have been discussed here. We would like to stress the view that the program does not generate data in the conventional sense, but rather segments of subgoal recognition program.

Finally, to counterbalance the simplicity of the examples treated on pages 83-86, Fig. 11 gives subgoal descriptions (together with graphic mnemonics) which present a typical list generated by program over a sequence of played games. It should perhaps be added that, with this list, the program plays at expert level.

REFERENCE

Elcock, E. W., & Murray, A. M. (1967). Experiments with a learning component in a Go-Moku playing program. *Machine Intelligence* 1, N. L. Collins and D. Michie (eds.), pp. 87-103. Edinburgh: Oliver and Boyd.

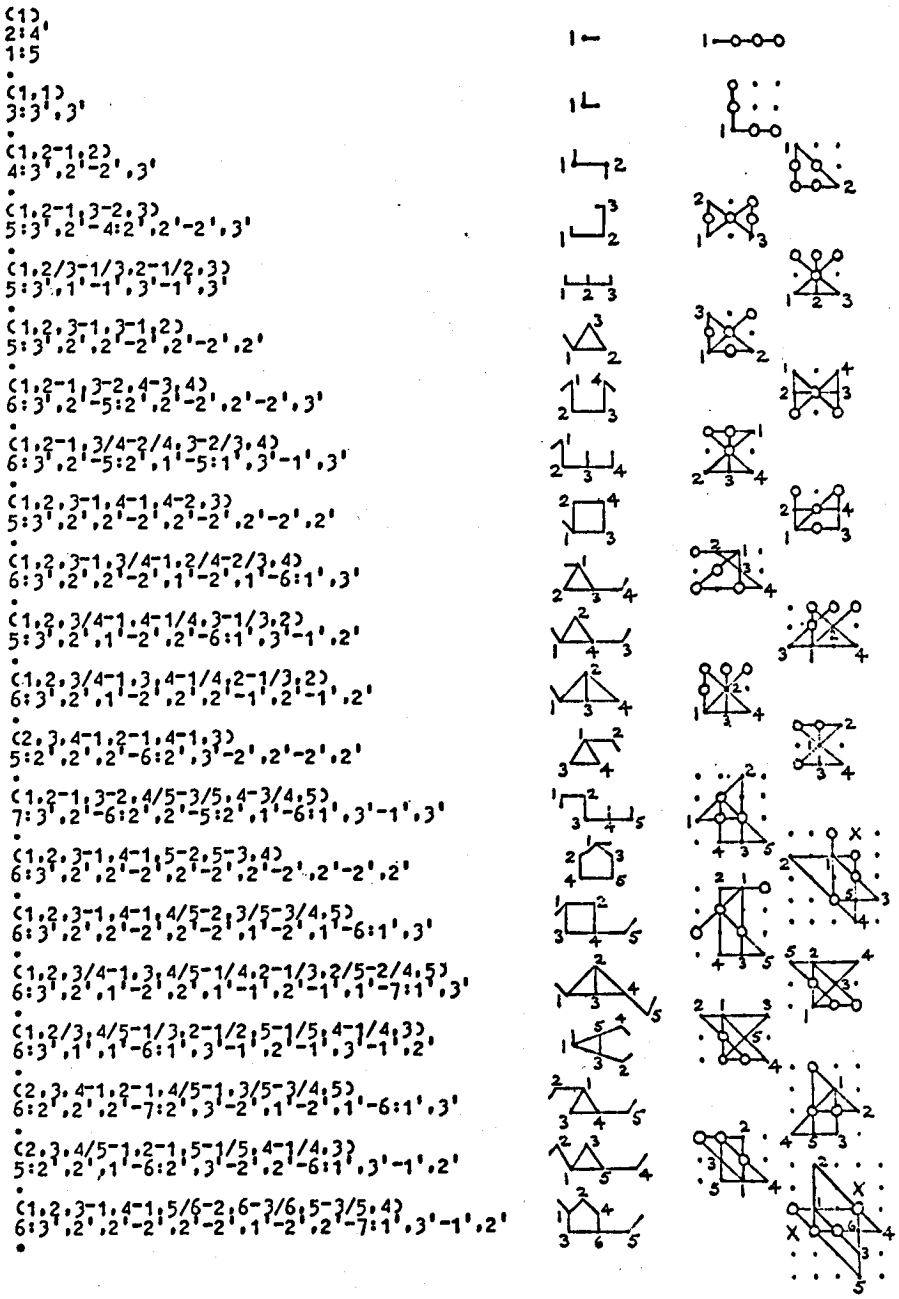

(1)
2:4'
1:5
(1,1)
3:3',3'
(1,2-1,2)
4:3',2'-2',3'
(1,2-1,3-2,3)
5:3',2'-4:2',2'-2',3'
(1,2/3-1/3,2-1/2,3)
5:3',1'-1',3'-1',3'
(1,2,3-1,3-1,2)
5:3',2',2'-2',2'-2',2'
(1,2-1,3-2,4-3,4)
6:3',2'-5:2',2'-2',2'-2',3'
(1,2-1,3/4-2/4,3-2/3,4)
6:3',2'-5:2',1'-5:1',3'-1',3'
(1,2,3-1,4-1,4-2,3)
5:3',2',2'-2',2'-2',2'-2',2'
(1,2,3-1,3/4-1,2/4-2/3,4)
6:3',2',2'-2',1'-2',1'-6:1',3'
(1,2,3/4-1,4-1/4,3-1/3,2)
5:3',2',1'-2',2'-6:1',3'-1',2'
(1,2,3/4-1,3,4-1/4,2-1/3,2)
6:3',2',1'-2',2',2'-1',2'-1',2'
(2,3,4-1,2-1,4-1,3)
5:2',2',2'-6:2',3'-2',2'-2',2'
(1,2-1,3-2,4/5-3/5,4-3/4,5)
7:3',2'-6:2',2'-5:2',1'-6:1',3'-1',3'
(1,2,3-1,4-1,5-2,5-3,4)
6:3',2',2'-2',2'-2',2'-2',2'-2',2'
(1,2,3-1,4-1,4/5-2,3/5-3/4,5)
6:3',2',2'-2',2'-2',1'-2',1'-6:1',3'
(1,2,3/4-1,3,4/5-1/4,2-1/3,2/5-2/4,5)
6:3',2',1'-2',2',1'-1',2'-1',1'-7:1',3'
(1,2/3,4/5-1/3,2-1/2,5-1/5,4-1/4,3)
6:3',1',1'-6:1',3'-1',2'-1',3'-1',2'
(2,3,4-1,2-1,4/5-1,3/5-3/4,5)
6:2',2',2'-7:2',3'-2',1'-2',1'-6:1',3'
(2,3,4/5-1,2-1,5-1/5,4-1/4,3)
5:2',2',1'-6:2',3'-2',2'-6:1',3'-1',2'
(1,2,3-1,4-1,5/6-2,6-3/6,5-3/5,4)
6:3',2',2'-2',2'-2',1'-2',2'-7:1',3'-1',2'

FIG. 11