

HIERARCHICAL REASONING IN THE GAME OF GO

David J H Brown
Computer Science Department
Teesside Polytechnic
Middlesbrough, Cleveland, England

The use of a semantic network to describe spatial and functional patterns at various levels of abstraction is described. Particular attention is paid to the use of a network by a deductive problem solver in forming plans and generating moves in playing Go.

1. PATTERN HIERARCHIES

It has often been remarked of Go that it is essentially a visually oriented game: such terms as "good shape", "thickness" and "eye space" abound in the literature. Imagery plays an important role in the conceptualisation of a board position.

RAG - an ongoing effort to program Go by integrating plan formation techniques with a multilevel pattern organisation - represents Go images by means of a semantic network [5]. The nodes of the network denote

- (a) organisational units such as points, strings, groups and territorial frameworks and
- (b) relations between units in terms of their functional roles (support, influence, etc.) and topology (linkages, cuts, pincers, etc.). Relations are themselves units and as such may be the parameters of other relations. For example, "diagonal" is a linkage relation which can exist between two strings and "cut" is a relation which can exist between a string and a linkage relation.

The natural hierarchy and taxonomy of such units is embodied in the network by means of subset, element and part-of relations. In effect, the network provides a continuum of abstraction spaces in which relations are permitted to exist between units at different levels of abstraction.

Fig.2 illustrates part of RAG's representation of the initial situation depicted in Fig.1. The network is implemented using the property-list structure of LISP. Nodes depicted by rectangles are pre-defined atoms; others (such as "BS2" which denotes the string of black stones at Q4, Q3, R3, S3 and S2) are created

dynamically. In the present implementation, network updates (as moves are played) are made on an ad hoc basis.

2. AN APPROACH TO REASONING

Because of the standardised representation provided by semantic nets, a uniform reasoning procedure may be employed to perform hierarchical planning. The rule-based inference system employed by RAG [1] is such a procedure. It utilises a set of packages of rules, each package containing rules likely to be applicable to a given class of goals. Packaging of rules is one way of obtaining plausible rules without the need to perform partial matches. This is especially important when rule matching is expensive; RAG's rules are represented as semantic nets, thus the matching problem is one of finding graph injections.

Each rule comprises a CONDITION, an ACTION and a GOAL. In forming plans, RAG matches its current goal with the GOAL of a rule. The CONDITION specifies contextual constraints under which the rule's ACTION should satisfy the instantiated GOAL.

In similar spirit to NOAH [4], RAG searches at the top level (subgoal on CONDITIONS) before proceeding to lower levels (subgoal on ACTIONS). Further, only the temporally most proximate goal of a plan is searched below the current level. The rationale behind this is threefold: (1) It being implausible to consider all continuations, no "proof" could be found for the correctness of a play; (2) Actions which are likely to enable goals are valuable as threats even if counterplay is possible; (3) The inclusion of CONDITIONS in rules enables applicability requirements to be examined abstractly; although Go is

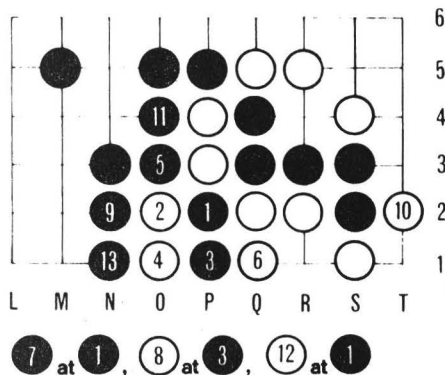


Figure 1: The Two-Stone Edge Squeeze

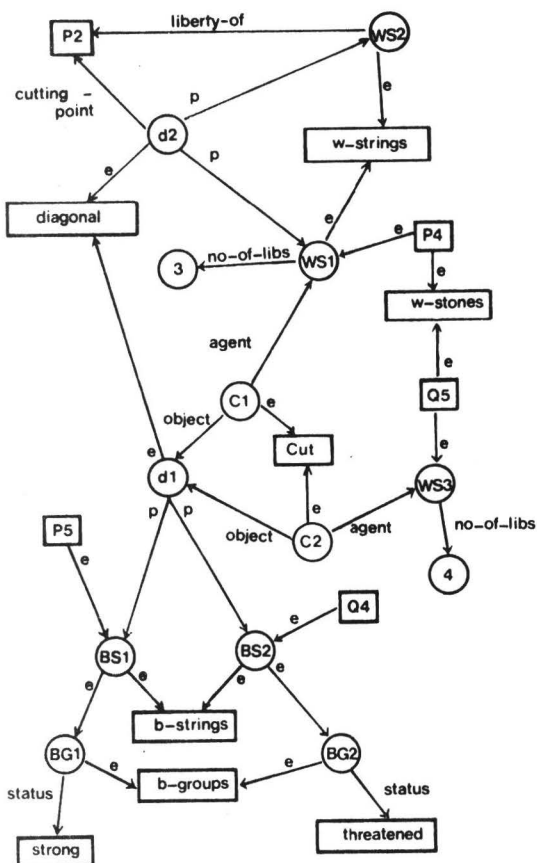


Figure 2: A Fragment of RAG's Representation of the Initial Situation (e=element, p=parameter)

theoretically a game of perfect information, it must be treated as one of imperfect knowledge as future events cannot be definitely predicted. Consequently, situations must be judged at an abstract level.

3. LOOKAHEAD

The effectiveness of RAG's rules cannot be guaranteed. It is possible to represent the necessary conditions of a rule's applicability (by its GOAL) but its CONDITION can only partly specify the sufficient conditions, given that a practicable system cannot be enumerative.

At the move generation level, certain sequences can be specified for individual tactics. The representation of Go images by semantic nets enables sequences, such as that in Fig.1, to be specified in terms of generalised moves. In the example, the first move is specified as:

(CUT (DIAGONAL ?W1 ?W2))

When fitted to the local context, the point P2 is established as the place to play to cut the linkage between WS1 and WS2.

RAG stores generalised move sequences in a database in which each sequence (tree) is indexed by the particular tactic (low-level ACTION) to which it relates. The responsibility for determining the likely relevance of tactics lies with the planning component.

4. SPECIAL TACTICS (TESUJI)

Tesuji are tactical plays known to be effective in certain types of situation. Such situations are characterised by combinations of features and relations in a local context. However, the variety of configurations that can have any one such combination is so large that template matching of stone patterns is infeasible.

Furthermore, each tesuji is inextricably linked with a goal; typical goals being to capture small groups of enemy stones to link up or make eye-shape, isolating weak enemy groups by cutting through linkages and so forth.

Thus the use of stone patterns, as in Zobrist's program [6], suffers from two deficiencies:

- (1) Prohibitively many patterns are required (Zobrist used about 200, which enabled his program to cope with only very elementary tactics); and
- (2) stone patterns are too primitive to be related to strategic considerations, ie. goals.

Attempting to solve the first of these, Ryder [3] developed a set of recursive definitions for certain forced sequences such as snap-backs and ladders. However, because his program lacked any strategic direction, it wasted many moves defending cuts or making eye-shape when it was unnecessary. The program of Reitman and Wilcox [2], with its multiple perceptual components, comes closest to the solution, but as yet suffers from a lack of strategic direction.

RAG is an attempt to solve both problems in a systematic manner. The planning component utilises rules which suggest various tesuji as a way of achieving certain goals. Each tesuji is passed to a move generator, which attempts to fit generalised sequences indexed by the tesuji into the current position. If a sequence fits, it is expected to achieve the desired goal.

As an example, consider the squeeze tesuji illustrated in Fig.1. This occurs in a race to capture between two eyeless groups. By sacrificing three stones, black can keep ahead in the race and squeeze the life out of white's cutting stones. The generalised move sequence black has comprises a cut (1), descent (3), cut of the newly-created diagonal (5) and throw-in (7) followed by removal of white's liberties.

In the example, the stones at P4 and P3 (WS1 in Fig. 2) are marked as cutting the diagonal connection between the threatened black group in the corner and the strong one on the outside. Q5 and R5 (WS2) are similarly marked, but they have too many liberties for the squeeze to work. Included among rules for saving threatened groups is the "two stone edge squeeze". This rule includes in its CONDITION the requirements: (1) None of the enemy strings connected to the one to be captured (including those connected by transitivity) has more than three liberties; (2) The string to be captured has a cutting point on the second line, with the point below it also vacant; and (3) The cutting point is unprotected (ie. a cutting stone would not be captured immediately).

Whilst there are a few configurations also characterised by these features to which the squeeze tesuji is not appropriate, their testing serves to eliminate countless others. Fitting the aforementioned generalised sequence to the situation in question serves to determine whether and how the original goal can be achieved.

5. SUMMARY

Semantic nets are a powerful representational formalism and are particularly suited to the description of spatial and hierarchical patterns such as occur in the game of Go. The use of a uniform representation for all levels of descriptive information enables a single deductive procedure to operate effectively in both strategic and tactical decision making.

The combination of graphical pattern matching and generalised sequence fitting enables the testing of low-level tactical goals to be performed, thus providing a verification mechanism for goals formulated at higher levels of abstraction.

ACKNOWLEDGEMENTS

Stu Shapiro first suggested the use of semantic nets for RAG. Don Beal, Tom Mitchell and Jerry Schwartz provided useful insights.

REFERENCES

- [1] Brown, D J H "Reasoning About Games". AISB newsletter 32 (1978) 14-17.
- [2] Reitman, W and Wilcox, B "Pattern Recognition and Pattern-Directed Inference in a Program for Playing Go". In Waterman, D and Hayes-Roth, F (Eds) *Pattern-Directed Inference Systems*. Academic Press, 1978, 503-523.
- [3] Ryder, K J "Heuristic Analysis of Large Trees as Generated in the Game of Go". AIM-271, Stanford University, 1972.
- [4] Sacerdoti, E "A Structure for Plans and Behaviour". AITN-109, SRI International, 1975.
- [5] Shapiro, S C "A Net Structure for Semantic Information Storage, Deduction and Retrieval". In *Proc. IJCAI-71*. The British Computer Society, 1971, 512-523.
- [6] Zobrist, A R "Feature Extraction and Representation for Pattern Recognition and the Game of Go". PhD Thesis, University of Wisconsin, 1970.