An Overview of ANDES: A Knowledge-based Scene Analysis System

Paulo Ouvera Simoni
Instituto de Pesquisas Espaciais - INPE/MCT
Av. dos Astronautas, 1758
12201 São José dos Campos - SP - BRAZIL

ABSTRACT

The representation and control components of the ANDES system are described along with some developed tests that exhibit their capabilities. A particular type of associative network is used, where predicate calculus-like formulas and decision rules share a uniform notation. Decision rules may be used in both forward, backward or mixed direction. Procedural knowledge may be given both as heuristics and as attribute, predicate or concept associated codes. The control component performs a series of propagation of alterations, problem solving and image measurements steps, where the latter may be executed either under request during an inference or in a spontaneous fashion. The notion of a recursive recognition cycle is present in the model as a precompiled feature, modifiable by heuristics or control options. There is also some provision for eliminating the effects of past inferences whenever an inconsistency is detected. The tests allowed to check the rule chaining process, the repetition of inferences and the mixed initiative character of the symbolic and image related processes.

1. INTRODUCTION

In recent years, many systems have been built for scene analysis and general vision tasks, incorporating mechanisms for explicitly representing available knowledge about the domain, scene and processes [3], [5], [6], [7], [8], [9], [10]. The ANDES (Associative Network and Decision Rule for Scene analysis) system [13] was designed by the author with the purpose of having a great flexibility for the knowledge representation and of having many predefined features in the control module (where the inference engine is included).

The particular kind of associative network that was implemented in this system has a propositional part in addition to the part and particularization hierarchies (see [2] for some models of associative networks). The syntax of decision rules, which will be exemplified along the paper, is an extension of that one usually defined [1], [12]. It allows one to write complex quantified expressions in premise or conclusion. Many algorithms used in the inference processes are common to both formulas and rules, due to the uniformity in the syntax of rules and formulas.

Declarative and procedural knowledge are used by the control component, allowing top-down, bottom-up or mixed recognition - the precompiled recognition cycle having multiple activation possibilities and optional

2. THE KNOWLEDGE REPRESENTATION MODEL

2.1. An example

The following statements (1)-(7) are an example of how part of the available knowledge about external scenes may be expressed in the implemented representation. Figure 1 shows the part and particularization hierarchies for (1)-(4). As part of the inheritance mechanism (see Section 2.3), the instances of TWO-STORY-HOUSE will also have the parts assigned to HOUSE; furthermore, the inherited parts may be referenced in a formula (as in (6)) or in a decision rule.

FLOOR, PART-OF HOUSE
(1)

WINDOW, PART-OF FLOOR
(2)

ROOF, PART-OF HOUSE
(3)

TWO-STORY-HOUSE, PARTICULARIZATION-OF HOUSE
(4)

VxVyWz [{ [HOUSE(x) AND WINDOW(x, y) AND
ROOF(x, z) ] ---> ABOVE(z, y)], [x] }
(5)

Vx [{ HOUSE(x) ---> [ TWO-STORY-HOUSE(x) ] --->
[y] [FLOOR(x, y)] ] }
(6)

Vz [{ [ACTIVE(z)] AND
[ {'z'} [ROAD(x) AND HOUSE(y) AND
ADJACENT(x, z) AND ADJACENT(y, z)]
AND INSIDE (AVE-COLOR(1, z), (120, 203)) ]

== (0.0) GRASS(z) ]
(7)

Predicates may be noncomputable or computable, and the latter need an associated code to compute their truth-values; geometric relation predicates are a subclass of this class. In this example, ABOVE is a geometric relation predicate; it checks if the center of mass of the region in the image given by the referent of z lies above the center of mass of that one given by y. This operation on the image can be executed only when all of its arguments have already been assigned known referents. ADJACENT is a geometric relation predicate too. INSIDE is a computable predicate that checks if the average brightness color component (computed by attribute AVE-COLOR) in channel 1 of a region given by the referent of z lies inside a specified interval of gray levels (green region, in this case). ACTIVE is a noncomputable predicate which has an initial TRUE value (with certainty 1.0) assigned to it when a region is created.

HOUSE(x) in formula (5) stands for a concept used predicatively. WINDOW(x, y) in formula (5) stands for a part relation between the concept designated by x and an instance of WINDOW. In general, this relation can be verified if there is a chain of part relations between a global concept and a part, and also a chain of the
corresponding instances of concepts and part relations. In this example, the stated relations (1)-(2) show such a chain between the generic concepts HOUSE and WINDOW and that there must be a corresponding chain among the already instantiated concepts, when (5) is used.

![Diagram showing hierarchies for example.](image)

**Fig.1 - Hierarchies for example.**

The use of part relation nodes in decision rules provides a way for explicitly stating the conditions under which a rule can be applied. In other systems, as in [12], the conditions are implicit, given by the context tree. On the other hand, it is acceptable to have hierarchies with more than one chain between two concepts.

Premise and conclusion of decision rules may have a common set of quantifiers (there is only one common quantifier in rule (7)). The first quantifier must be universal; other arbitrary sets of quantifiers may appear inside either the premise or the conclusion. The use of at least one most external common universal quantifier comes from the idea of providing a way for specifying a common "context" for the concepts and predicates that appear in a decision rule. Certainty factors act over all the terms of the conclusion (a more complex case than (7) is (20), in Section 4.2).

Rule (7) may be used for propagating the ACTIVE condition, as well as for answering an inquiry either about the existence of a region whose identification is grass or about the identification of a specific region as being grass.

### 2.2. Additional features

Other features of the representation, not shown in this example, are:

- The concepts may have an associated code to be executed when they are instantiated (to activate chosen portions of the part or particularization hierarchies, for instance). As a comparison, the type of representation used in [9], a semantic net with provision for procedural knowledge attached to concepts, can be partially simulated in the ANDES system, due to the necessary parts: these parts should be stated as such by means of formulas or rules.
- Predicates have an indication if the computation of their truth-values requires measurements on the image. For computable predicates, the order of the first argument that has its referent assigned during evaluation may be specified also. This feature allows the assignment of a value coming from an attribute or from a function to a variable.
- The specification of attributes includes the type of their values (number, interval or list), associated code for computing their values and indication of automatic computation for new regions that are created.

### 2.3. Inference of part relations

Part relations may be inferred as a result of the inference rules (8)-(10), which are applied automatically whenever a formula or a decision rule with part relation nodes is used. The inference of a relation is inhibited if the inverse relation is explicitly stated.

\[
\begin{align*}
&\text{If } \text{B PART-OF A and C PART-OF B,} \\
&\quad \text{then C PART-OF A} \\
&(8)
\end{align*}
\]

\[
\begin{align*}
&\text{If } \text{C PART-OF A, B PARTICULARIZATION-OF A and} \\
&\quad \text{B } \not\equiv \text{ C,} \\
&\quad \text{then C PART-OF B} \\
&(9)
\end{align*}
\]

\[
\begin{align*}
&\text{If } \text{B PART-OF A and C PARTICULARIZATION-OF B,} \\
&\quad \text{then C PART-OF A} \\
&(10)
\end{align*}
\]

The algorithm for applying these rules (not shown here; see [13]) is able to find an (inferred or original) relation or chain of relations between two concept nodes whenever there is at least one; it expands the part hierarchy by applying (8)-(10) recursively.

Inference rules (8) and (9) implement the notions of transitivity of part relations and property inheritance for part relations, respectively. Rule (10) is useful in several contexts. They allow some degree of hierarchy compression. The expanded hierarchy is stored so that new references to the same formula or rule do not cause a new part inference process.

As an example, if (5) is to be activated for inference, an auxiliary graph with nodes HOUSE, WINDOW and ROOF is built and a path is sought between HOUSE and WINDOW nodes (due to the common variable x) and also between HOUSE and ROOF nodes (due to x). For the first path, FLOOR is added to the graph, along with the relations FLOOR PART-OF HOUSE and ROOF PART-OF HOUSE; FLOOR is chosen as next node (no one of the sons of the HOUSE node is the sought node); a relation is found between FLOOR and WINDOW and it is added to the graph.

When using (6), a path is sought between HOUSE and FLOOR nodes and also between TWO-STORY-HOUSE and FLOOR nodes. For the second path, HOUSE node is chosen to be traversed as a generalization of TWO-STORY-HOUSE; ROOF node is stored, as well as the relations FLOOR PART-OF HOUSE and ROOF PART-OF HOUSE; TWO-STORY-HOUSE receives the parts ROOF and FLOOR, by (9).

### 3. THE CONTROL COMPONENT

#### 3.1. The structure of the system

Two separate processors, the symbolic and the image related, are simulated by the only one existing physical processor via time splitting and priority assignment. The former one is activated by inferences due to formulas or decision rules or by propagation of values through the hierarchies, and the latter one is activated either spontaneously or by calls coming from the codes associated with predicates and attributes, during the just mentioned inferences. The dynamic communication between these processors employs the control structures given by the lists of measurements, of alterations and of problems, shown in Figure 2.

The list of measurements acts as a stack, ordered by priority and by time; it stores awaiting requests of
measurements on the image. When a request comes from an inference process due to a formula or to a rule, the process is deferred until the request is fulfilled.

The hypothesis base stores the instantiated part of the associative network, and is composed of the activated concepts and their referents (pointers to the image description, for instance), part relations, attribute and predicate values and referents. It also stores the history of the truth-values associated to instances of concepts, in order to allow backtracking. The hypothesis base instances (in particular the interpretations associated to regions) may come from some measurement on the image, from inferences made by a rule or by a formula, from hierarchy modifications or from a tentative assignment. Formulas and decision rules already used for inference are also stored, together with the elements mapped to their nodes and the referents of the variables whose referents were set as fixed before the inference process began.

The list of alterations stores pointers to elements that were inserted in the hypothesis base or that had their values or truth-values modified. The control module may propagate them or question about their validity, resulting in the activation of formulas or decision rules. There are some predefined criteria for choosing the next alteration to be propagated: the first one, the last one or all of them. Other heuristic criteria may also be given.

The list of problems stores the concepts or predicates for which an instance is sought and for which some referents may be unknown. Problems arise either as an user query, as part of a heuristics, as a subproblem when a formula or rule is accessed or from the hierarchies related to a concept.

As an instance, if LIVES is a predicate symbol with four arguments (person, place, initial-time, end-time) and one wants to know who lived in a certain place PLACE-1 from 1984 to 1986, a problem "LIVES(PLACE-1, 1984, 1986)" would be generated. If the predicate LIVES can be matched to a node in the conclusion of a rule, it will be chained in a backward mode. If some node in the premise of this rule has no corresponding element in the hypothesis base or if all of the existing elements have unknown values, subproblems of the original problem are generated (see Section 41 for an example). These subproblems may have in general additional unknown referents, in relation to those of the initial problem. The solution of this problem is the set of all instances of LIVES that can be inferred from the stored knowledge using known facts, such that the known referents of the problem match the corresponding referents of the instances.

The 3-channel 256 gray level image is an input to the system. The image description comprises a set of structures generated from the image and from other structures. The formed regions, their masks, their historic status (coming from split, merge, etc.), the interpretations associated with them and codes associated to lines in the image are some of the structures. This set of structures is stored in distinct places from the remaining structures, so that extensions to other domains (speech or sequences of images, for instance) should require minor adaptations.

### 3.2 Some details about the control process

The general structure of the control cycle is outlined below. Several feedback links, internal to sets of actions, were removed for simplicity.

- Propagation of alterations and problem solving:
  - eliminate already propagated alterations;
  - determine actions to perform;
  - seek for suspended formulas or rules;
  - choose problem to solve;
  - choose alteration due to a tentatively altered value or due to an unknown value;
  - access formulas and rules.

- Preparation for inference:
  - choose mapping;
  - eliminate alterations;
  - compute direction indications.

- Inference:
  - use mapped formula or rule;
  - perform postinference analysis;
  - seek conditions for reactivating a formula or a rule.

- Measurements on the image:
  - choose next measurement;
  - set initialization actions.

This cycle goes on until there is no more alteration to propagate, problem to solve or measurement to perform, after having taken into account the available initialization actions. These actions may be chosen whenever there is no possible action or measurement at a certain level (attach a tentative interpretation to an uninterpreted entity, attach a tentative TRUE truth-value to interpretations with unknown truth-value are some of these actions). User-specified actions can also be used instead.

At the beginning of each cycle, an user specified routine may be used as a heuristics for selecting a specific problem to solve or an alteration to propagate, for initializing a list, etc. It is also available a set of control commands that alter the control flow, in order to choose the order of propagation, to activate the hierarchies of an instantiated concept, to define places for interruption, to choose among predefined initialization procedures, etc.

When accessing formulas or rules, their nodes must be mapped to the elements that triggered the access,
and may also be mapped to other nodes in the hypothesis base and to fictitious elements. Some of the universal variables may have referents attached to them, coming from the mapped elements; other variables, universal or existential, may require a search process over the hypothesis base during the inference process in order to verify the quantified expressions. A part relation inference process (as seen in Section 2.3) may be needed. Heuristics may be written for choosing the mapping to be used for the inference process. Some indications are associated to formula or rule nodes, according to the existence of a question, computable predicates, and other cases, in order to state if a truth-value will be assigned to the element mapped to a node or if a known value will be used.

A list of provisional values is built as a result of the inference process due to the use of a formula or decision rule. The elements of this list are inserted in the hypothesis base if the use of the formula or rule does not result in a conflict or if the process was not suspended due to a stacked measurement or problem. A hierarchy propagation may take place if a concept is instantiated or if its truth-value is modified. As a consequence of an inference process, if the concluded values are such that an inconsistency among truth-values is detected and it is chosen to discard one of the conflicting values, a backtracking process is invoked. During this process, other values inferred as a consequence of one of the conflicting values may also be recomputed, using the formulas or decision rules that generated these other values. This kind of behavior will be made more explicit in Section 4.2.

Some of the available operations on the image are segmentation, histogram thresholding, shape and color attribute computation and geometric relation computation.

Some additional comparisons can be made to other existing models. Glicksman's schema-based system [3] uses a perception cycle as the basic control mechanism (such as Havens' [4] system) and this kind of control can also be attained in ANDES: measurements on the image generate propagation of results, from which interpretations may be assigned, activating parts of the hierarchies; all these hypotheses will invoke related formulas or decision rules, which would ask for more measurements. The same amount of knowledge of a system for recognizing man-made objects [6] (contextual, domain specific and picture specific) could be represented, in different form, in ANDES: measurements on the image generate propagation of results, from which the mapping process is very simple in this case (where the rules do not have part relation nodes); due to the fact that the referent of the element corresponding to A(x) is unknown, a search over the hypothesis base is performed in order to find the corresponding elements for the other nodes with the same variable; this brings out B(1) and B(2); two mappings are generated, corresponding to B(1) and B(2), respectively; during the use of this rule, the problems "C(1) ?" and "C(2) ?" are stacked, one for each of the mappings (backward chaining of 11).

The executed actions are summarized as follows:

11) is accessed;
the mapping process is very simple in this case (where the rules do not have part relation nodes);
due to the fact that the referent of the element corresponding to A(x) is unknown, a search over the hypothesis base is performed in order to find the corresponding elements for the other nodes with the same variable;
this brings out B(1) and B(2); two mappings are generated, corresponding to B(1) and B(2), respectively;
during the use of this rule, the problems "C(1) ?" and "C(2) ?" are stacked, one for each of the mappings (backward chaining of 11);
the certainty factors are arbitrary. There is an initial problem represented by "A ?"; meaning that instances for the concept A are sought and they must be found by inferences from the above mentioned available rules.

\[ \begin{align*}
\text{11) is accessed; } & \text{the mapping process is very simple in this case (where the rules do not have part relation nodes); due to the} \\
& \text{fact that the referent of the element corresponding to } A(x) \text{ is unknown, a search over the hypothesis base is performed in order to find the} \\
& \text{corresponding elements for the other nodes with the same variable; this brings out } B(1) \text{ and } B(2); \text{ two mappings are generated, corresponding to } B(1) \\
& \text{and } B(2), \text{ respectively; during the use of this rule, the problems } "C(1) ?" \text{ and } "C(2) ?" \text{ are stacked, one for each of the mappings (backward chaining of 11);} \\
\end{align*} \]

A(1) ?:
11) is accessed and rejected (the question corresponds to a node in the premise);
12) is accessed and the truth-value of C(2) is computed;
13) is accessed and rejected;
C(1) ?:
the same rules are accessed and the truth-value of C(1) is computed;
after solving the above problems, the suspended inferences by the mappings of 11 are resumed and the truth-values of A(1) and A(2) are computed;
A(1) :
11) is accessed and rejected (it corresponds to a node in the conclusion);
C(1) :
11) and 12) are accessed and rejected;
13) is accessed and the truth-value of H(1) is computed (forward chaining of 13);
the truth-value of HG(1) is also computed, using the particularization hierarchy (16);
HG(1) :
14) is accessed, generating the problem "C(1) ?" (no instance for I is known up to this time);
As a comparison, several of the operations of [10] can be simulated through the decision rules and control mechanism; moreover, ANDES rules do not have predefined directions for rule chaining.

42. Nonmonotonicity

A test was designed in order to check the backtracking mechanism responsible for repeating the use of decision rules or formulas, when some condition that was valid at a previous time does not hold anymore.

For this purpose, a well-known algorithm (the interpretation-guided segmentation algorithm [14], in particular its constraint propagation method) was chosen to be reproduced. The potential applicability of this algorithm is not discussed here; it was aimed only to repeat the steps performed by it, using now a basically declarative specification of it. The computational efficiency of this new implementation is not discussed also; the objective was to show the correspondence of the backtracking process.

In order to attain this behavior, it was important to take into account all possible orderings of the constraints, due to the sequential nature of the implemented propagation process. It was also important to be able to generate the correct answer for all possible orderings of the pairs of regions of the image (the image chosen for the test was the same used in the mentioned report [14]).

Initially all regions were assigned all possible interpretations, in a tentative basis, and these interpretations were added to the list of alterations. Decision rules were used to specify the conditions under which interpretations could be rejected. The net effect of the inferences performed by them is equivalent to the effect of applying the constraints used in [14]. Some of the 19 rules used for this purpose are shown in (18)-(20), where B, F, K and P stand for Baseboard, Floor, Doorknob and Picture, respectively. The control process has enough flexibility to allow the negation of a condition that holds in the premise of a rule.

ABOVE and WITHIN are geometric relation predicates, and SMALL is a computable predicate. The noncomputable predicate ACTIVE was added so that a referent for the existential variable y could be found in a rule like (20). This is necessary because in this rule y was used only as argument of a computable predicate and the referent of the argument of this predicate needed to be determined before the truth-value of this predicate could be determined.

\[
\forall x \ [ \ [ P(x) \text{ AND NOT SMALL}(x) ] \\
\text{implies} \ (1.0) \text{ NOT } P(x) ] \\
\text{implies} \ (1.0) \text{ NOT } P(x) ] \\
\] (18)

\[
\forall y \ [ \ [ B(x) \text{ AND NOT F}(y) \text{ AND ABOVE}(x,y) ] \\
\text{implies} \ (1.0) \text{ NOT } B(x) ] \\
\text{implies} \ (1.0) \text{ NOT } B(x) ] \\
\] (19)

The mentioned set of decision rules simulated the constraint propagation process under comparison. The inconsistency analysis, called in the postinference phase whenever a value or truth-value is modified in the hypothesis base, had a special purpose procedure attached to it for accepting a new tentative value and for forgetting the old values in the list of alterations and in the hypothesis base.

The initial interpretations were propagated as known data and mapped to premise nodes with universal variables. The interpretation for region R1 propagated initially, B(R1) activates (19), among other rules, and this rule is rejected (there is no y such that \( F(y) \) is \( \text{FALSE} \) at this time). B(R1) also activates (20), which results in B(R1), F(R1), K(R1) and P(R1) with truth-value \( \text{FALSE} \). All of the rules such that one of these instances was mapped to a concept node with universal variable argument (the referent of the variable was fixed for these inferences) and such that they were used in a time subsequent to the former setting (initial setting, in this case) of the truth-values were reactivated for a new inference, due to the fact that the truth-value of these instances changed from the value that holded for the former inference. Among the rules, (19) is activated again, but no new inference results from it. The process went on until there was no more alteration to be propagated, and the final interpretations were the expected ones.

It was observed during the test that the algorithm for accessing formulas and rules was the most time consuming. It was also observed a great number of tentatives to reuse already used rules, whenever an interpretation was rejected.

Another set of 56 rules was necessary in order to reproduce the merge process of [14] and the new constraints for this situation. It was stated that, if some geometric relation holds between regions R1 and R2, then there is a set of possible interpretations for R2 corresponding to each interpretation of R1.

43. Separation of processors

Another test exploited the semidependent nature of the image processor. A histogram-based split process was called either spontaneously or by explicit reference to it in nodes of decision rules. Some of the rules are shown at (21)-(24); they were written to assign interpretations with certainty factors (water, vegetation, ...) to regions (21), to impose conditions on the call to the above mentioned process (22)-(23), and to perform the actual call (24). The contextual conditions comprised the cases of regions with the gray-level variance above a given threshold (22), of regions with no interpretation assigned to them (23) and of regions with more than one valid interpretation (not shown here).

\[
\forall x \ [ \ [ \text{ACTIVE}(x) \text{ AND} \\
\text{INSIDE}(\text{AVE-COLOR}(x,1), (2, 32)) ] \\
\text{implies} \ (0.8) \text{ WATER}(x) ] \\
\text{implies} \ (0.8) \text{ WATER}(x) ] \\
\] (21)

\[
\forall x \ [ \ [ \text{VEGETATION}(x) \text{ AND NOT DOONE_SPLIT}(x) \text{ AND} \\
\text{NOT POTENTIAL_SPLIT}(x) \text{ AND} \\
\text{GREATER-THAN}(\text{VARIANCE}(x), 50) ] \\
\text{implies} \ (0.8) \text{ WATER}(x) ] \\
\text{implies} \ (0.8) \text{ WATER}(x) ] \\
\] (22)
POTENTIALSPLIT is a noncomputable predicate which has a tentative FALSE truth-value assigned to its instances whenever a region is formed. DONE_SPLIT is also a noncomputable predicate with the same initial value setting. SPLITREGION is a computable predicate that has as associated code a call to the routine that performs the split operation (it computes the histogram for the region under attention, chooses the most promising peak in the histogram, computes the masks of the subregions corresponding to this peak and to the remaining part of the region and separates the region in connected components; no attribute is automatically computed for the subregions). This code also performs the initial assignment of ACTIVE, DONE_SPLIT and POTENTIALSPLIT to the new subregions (these assignments are not explicit in the declarative part of the decision rules, as it should be desirable, due to the fact that the number of new regions is not known a priori and they cannot appear as arguments of the predicate).

The image used in this test was a 64x64 256 gray-level artificially generated image, with ten regions whose average gray-levels were in accordance with water, vegetation, and land interpretations. Initially the list of alterations had a pointer to the instances of ACTIVE(R1), POTENTIALSPLIT(R1) and DONE_SPLIT, where R1 stands for the whole image. When ACTIVE(R1) was propagated, it matched the premise of (21)-like rules and of (23). Measurements on the image (AVE-COLOR) were needed for the (21)-like rules and they were suspended. Rule (23) accessed the code of the attribute NUMBER-INTERP, and an instance of POTENTIALSPLIT(R1) was added with value TRUE; this new value was compared in the postinference step of the process with the older one and this latter was forgotten, by means of a heuristics for this problem.

At this point the image processor took control (according to the time balance between processors that was chosen for the test); there were some awaiting requests for measurements. The pending actions were executed (AVE-COLOR attribute was computed). The symbolic processor was in duty again and the suspended rules were reactivated; the (21)-like rules generated no assignment of interpretations. The instance of POTENTIALSPLIT was propagated, (24) was activated and a call to SPLITREGION performed, creating five new subregions. The predicate instances for these subregions were propagated. At the end of the process the image was separated in the specified regions.

When started with another time balance between processors (more favorable to the image processor and also to spontaneous operations), a spontaneous call to the split operation (the only image operation available for spontaneous calls, in this example) was performed.

5. CONCLUSIONS AND EXTENSIONS

The implemented system presents a number of features in the representation and in the control module that were used in order to attain a greater flexibility. These features lie among those mentioned by Rosenfeld [11] in his discussion about "expert" vision systems. Some drawbacks were observed, concerning the time-consumption in some parts of the program and the need for a command language with some predefined options for the specification of the heuristics. Some intended improvements are the simulation of the behavior of some expert systems [1], [12] and the expansion of the available image structures. Another desirable situation in which the part relation inference could be used is during hierarchy acquisision, in order to avoid cycles; in the current version of the system, the cycles that can be detected are those containing only original part relations.

ACKNOWLEDGMENTS

The author wishes to thank his thesis advisor Dr. Celso de Renna e Souza for his suggestions and encouragement during the work that resulted in this paper.

REFERENCES