Meta-Reasoning for Multi-Spectral Satellite Image Interpretation

Paul Robertson and Robert Laddaga

BBN Technologies
10 Moulton Street
Cambridge, MA 02138
USA

Abstract

The role of meta-reasoning in a satellite image interpretation program is described. A brief description of the satellite image understanding program is given, as well as a description of the self-adaptive architecture GRAVA. Meta-knowledge and abstract knowledge are shown to enable self-adaptation which results in greater robustness and reliability in the program.

Introduction

In this paper we discuss meta-reasoning, in an image interpretation architecture called GRAVA, and its role in producing good image interpretations under a wide range of environmental conditions.

In Metareasoning: a manifesto (Cox, Raja 2007) discuss introspective monitoring. Online monitoring is an essential attribute of any system that operates in an uncertain environment. Online monitoring allows failed approaches to be replaced by alternatives, contingent plans to be applied when available or to invoke failover procedures to be invoked. Such recovery mechanisms are fruitless when the failure suggests that the entire approach has been invalidated. Metareasoning is essential to be able to reason about the validity of the current approach to avoid having the program fruitlessly apply contingent methods in an environment where they will have no beneficial effect – and may lead to disastrous results. Nowhere is this need more apparent than in computer vision applications where changes in illumination, image context, and a host of other environmental factors can demand a completely different approach to the image understanding process. Most successful computer vision programs are is situations where conditions can be carefully controlled. Computer vision systems on mobile robots forces vision into uncontrollable environments and we argue that metareasoning the cleanest path to reliable systems in this domain. Our approach draw upon observations by Lessor et.al (Erman et.al. 1980) on the Hearsay II project but are implemented in an architecture that is self-adaptive (Robertson 2002). Metareasoning in our approach takes the form of reasoning about the *performance* of a subordinate computational process. Performance measurements can come from two sources: specific failures invoked by the subordinate process and poor performance in comparison to learned performance models. We utilize description length as one important component of learned performance models. Metareasoning in our system operates on *contexts* that are learned with the help of a clustering algorithm (Robertson, Laddaga, 2002).

We consider a program that interprets a sequence of satellite images formed from the passage of the satellite as it traverses its orbit. The sequence of images thus created will depict the changing landscape visible from the directed camera of the satellite. These landscape changes we will treat as one component[1] of the environment of the satellite image interpretation program.

The real world is complex and attempting to build static vision programs with hardwired logic to deal with every possibility is naive. The world in which we live exposes the eye, and camera, to an enormous diversity of visual variety. When faced with complexity, it is often a good strategy to 'divide and conquer'. We divide by contexts, and conquer each context with an individually tailored program configuration.

Consider a collection of programs that specialize in interpreting different scene types and some way of switching between programs as necessary. Not only would such an approach make the complexity seem more manageable but the constraints of the scene would allow better interpretations to be generated. The correct identification of objects in a scene may not be uniquely discernable from the pixels in the image, but the ambiguities might be resolvable from the context in which they appear. For example, a program

[1] Other components of the environment include the type of camera or sensor, and the weather and atmospheric conditions.

that has to deal with ocean scenes is likely to correctly interpret large tankers as ships whereas similarly appearing structures in an industrial scene would be interpreted as, say, a warehouse.

GRAVA employs a self-adaptive approach towards achieving reliable interpretations in complex and changing environments. Self-adaptive software is concerned with the problem of automatically adapting program *structure* in response to environmental changes in order to provide robust performance despite those changes. Self adaptation encapsulates the idea that a running program may be viewed as a collection of context dependent programs that are automatically selected in sequence as context changes are detected. The selected program is assumed to have the property that it is a good match for the environment at the time of the adaptation, but this assumption is continuously checked.

The satellite image interpretation program described in this paper segments, labels, and parses aerial images so as to generate a rich structural description of the image contents for each image in the sequence.

Even if we could know all the different states that the environment could be in, we wouldn't know *a priori* what state the environment would be in at any particular time. Consequently, in order to achieve robust performance, image understanding programs should determine the state of the environment at runtime and adapt accordingly.

Self adaptation is a model-based approach to building robust systems. The environment, the program's goal, and the program's computational structure are modeled. In principle, the idea is simple. The environment model and the program goal model support continuous evaluation of the performance of the program. When program performance deteriorates, the program goal model and the computation model together support modification of the program structure. In this way, the program structure evolves as the environment changes so that the components of the program are always well suited to the environment in which they are running. Robust performance results from having all components operating within their effective range.

GRAVA employs meta-reasoning because it provides the mechanisms necessary to support two of the core problems of self adaptive software—a mechanism for reasoning about the state of the computational system and a mechanism for making changes to it.

While the focus of this paper is on the application of meta-reasoning in producing reliable image descriptions in the satellite image understanding program, a brief overview of the approach taken by the program in interpreting visual scenes is helpful.

To produce an image interpretation, a variety of tools are brought into play. Tools vary both in terms of the goals they are intended to achieve, and by the methods they use. Each particular goal may be satisfied by many different algorithms or implementations. The selection of the right tools determines ultimately the quality of the resulting interpretation. First, the image is processed to extract features such as texture by applying tools appropriate to the image at hand. Next, a segmentation algorithm is employed in order to produce regions with outlines whose contents are homogeneous with respect to content as determined by the chosen texture and feature tools. The segmentation algorithm also depends upon tools that select seed points that initialize the segmentation. The choice of tools to initiate the segmentation determines what kind of segmentation will be produced. Next the regions of the image are labeled based on their contents. Finally, the image is statistically parsed using a 2D picture grammar.

At any point, a bad choice of tool—for initial feature extraction, seed point identification, region identification, or parse rule—can lead to a poor image interpretation. The earlier the error occurs, the worse the resulting interpretation is likely to be. For example, a poor choice of tools for extracting textures from the image may result in a poor segmentation. The poor segmentation will likely result in poor region content analysis and a potentially disastrous parse of the image. The resulting interpretation can be very bad indeed. The challenge for a self-adaptive approach is to determine how the program formed from the collection of tools described above can be organized so that when a poor interpretation is produced, the program self adapts into a program that does a better job.

Meta-Reasoning

Before delving into the details of meta-reasoning in GRAVA we begin with some general explanation and motivation for meta-reasoning.

Meta-reasoning involves reasoning about the reasoning process. Meta-reasoning is just like reasoning but it is applied to the reasoning engine rather than to the problem domain. Similar results could usually be achieved by adding additional logic to the reasoning system but several advantages accrue by having the meta-reasoning be structured as a separate layer of the reasoning process:

1. **Code cleanliness**: Mixing meta-reasoning in with the reasoning logic is bad modularity. It makes the reasoning code hard to read, maintain, and debug.
2. **Synchronization**: Meta-reasoning may want to be invoked at different times to the base reasoner. Putting them together makes this very difficult.
3. **Representational efficiency**: By reasoning at the meta-level a whole class of issues can be handled

in one place that would have to be repeated for each member of the class at the reasoning level.

For meta-reasoning to take place the base reasoner is instrumented to allow the state of reasoning to be reasoned about. For meta-reasoning to influence the reasoning process the results of meta-reasoning are reified into the reasoning process. This can be accomplished by representing the results of meta-reasoning as data structures that are used by the reasoner or by interrupting the reasoning process and restarting it with new structures.

Meta-reasoning in GRAVA not only reasons about the reasoning process but it can interrupt the reasoning process and redirect it in new directions (as will be demonstrated below).

Meta-levels in GRAVA

The GRAVA architecture was designed to support image interpretation programs that take advantage of self-adaptation in order to effectively handle a changing environment. Self adaptation results from a meta-reasoning event (also known as reflection). In GRAVA, an application is structured as a hierarchy of processing levels. Base-level reasoners reason about the image that they are attempting to interpret. Meta reasoning can occur in any of the levels above a base reasoner. To understand how meta-reasoning in GRAVA works we begin by giving a brief overview of salient aspects of the GRAVA architecture.

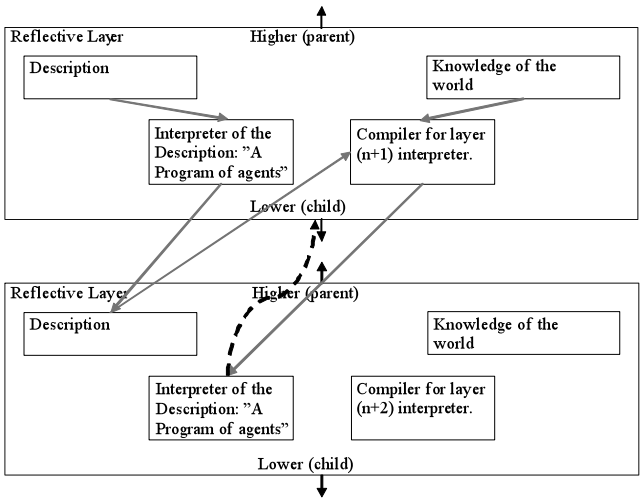

Figure 1: The GRAVA architecture

Figure 1 shows two levels of a GRAVA program. Each layer of a GRAVA program contains a *description* that specifies the goal for the layer. The implementation of that description is a program consisting of a collection of agents (the tools) that is fixed at the top level but which is synthesized by the parent layer for all lower layers. The lowest level interprets the image by applying the agents that make up the interpreter. The parent layer of each layer (except for the top layer) contains a compiler whose purpose it is to synthesize the interpreter for the layer below by considering two sources of information:

1. The description for the child layer.
2. The knowledge of the world as collected by the child layer.

The "knowledge of the world" is the result of reasoning about the child layer and the product of the child layer. Since the knowledge of the world is an input to the compiler of the child interpreter it indirectly affects the reasoning of the child process.

Reasoning in GRAVA

GRAVA reasons about image contents by fitting models to parts of the image. This process is achieved as part of a bidding process among competing agents that each attempt to explain image parts. The winner is selected by a Monte-Carlo selection mechanism that uses description length of the competing representations to bias the selection. If some area (x) of an image fits the agent 'airport' and is selected by the Monte-Carlo selection this means that the reasoner has executed the rule:

$$area(x) \ \& \ airportAgent \rightarrow airport(x)$$

Levels of Interpretation

The satellite image interpretation program has three levels as follows.

Level 3: The bottom level interprets the pixels of the image as non-overlapping regions. The regions are obtained by a semantic segmentation algorithm (Robertson, P. 2002).
Level 2: The Segments are labeled with semantic markers.
Level 1: The Labeled Segments are parsed using a picture grammar (Robertson P. 2002).

The parse tree produced by level 1 has segments as its leaves which in turn have labels indicating their interpretation as objects in the scene. Monte-Carlo sampling occurs across all levels so that segmentation, labeling, and parsing choices are made so as to achieve the final parsed image with a minimal description length (MDL). The Monte-Carlo sampling employed to achieve this result is an anytime algorithm so after a single iteration an image description is available but much of the description will likely be incorrect. After multiple iterations the best (MDL) description emerges.

Model Learning and Contexts

Image interpretation with GRAVA is a two phase process. The first, off-line, process learns models from hand annotated training data—ground truth. These models include:

1. Grammar rules that model the 2D picture structures found in the annotated images;
2. Labeling models that map region contents to the named labels annotated by the human annotator and used in the grammatical rules as terminal symbols; and
3. Region content models that determine the features used to drive the segmenter.

These models correspond to the three layers of the aerial image interpretation program described above. The training images are clustered[2] to produce model groups that correspond to different contexts. This is done for each of the levels.

The resulting clusters may distinguish different imaging contexts. For example, in our training set, images from different multispectral bands were provided. At the label level, images are clustered based on different labels present in the images, and at the grammar level images are clustered based on the grammatical forms present in the images.

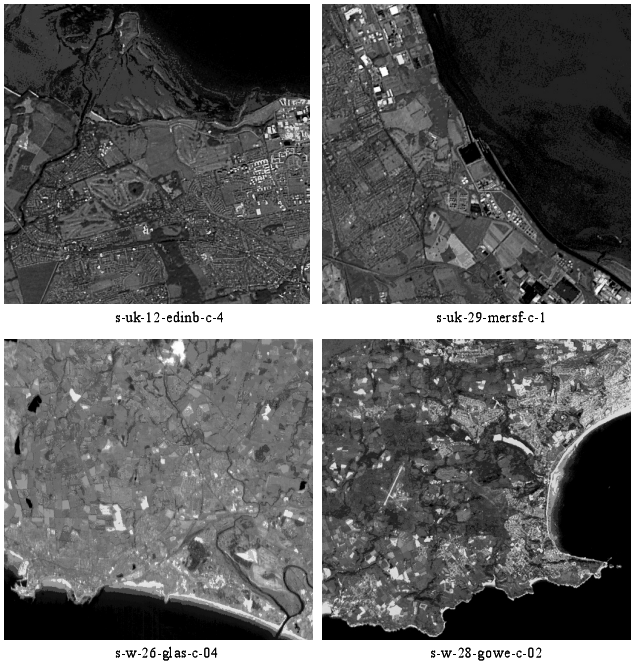

Figure 2: Four images from the same context

While the contexts may not correspond to simple English descriptions they do represent different imaging features that require different tool sets. Selecting a context thus constrains how the image will be interpreted, by determining the tool set.

Figure 2 shows four images that were clustered together into the same context by virtue of their similar label sets. One of the four images (bottom left) clearly comes from a different multispectral camera so these images would not have been clustered together at the lower level that clusters on the basis of optical information content. Clusters, of course, are not given names and may not even obviously be related. These images however all appear to be good examples of sea towns. Selecting agents based on the correct context dramatically improves the likelihood that the resulting image interpretation will be accurate.

Each of the contexts in each of the categories comes with estimated *prior* probabilities based upon their frequency in the training set. In the absence of any other information these priors allow the contexts at each level to be sorted in order of decreasing likelihood. The contexts at each level are used by the compiler to drive the selection of agents for each corresponding level of the program.

The Role of Contexts in Self-Adaptation

Meta-reasoning monitors the operation of the reasoner and estimates the most likely context based on the current state of knowledge about the world and the child layer's process state. When it determines that the currently selected context is not the most likely it re-synthesizes the program operating at the level beneath it to utilize agents appropriate to the newly selected context. At the parsing level this may mean that a set of grammatical rules are replaced by another, more appropriate, grammar. For example, if the grammar set was for a class of 'rural landscape' images and collected world knowledge suggests that 'seaport town' images represents a more appropriate context, the agents (grammatical rules) for a seaport town will be used in place of those for rural landscape. This will enable the seaport town to be properly parsed where a correct parse would have been impossible with a rule set dedicated to rural landscapes.

Reasoning and Meta-Reasoning about Context

While the above sketches out how, for the satellite image interpretation domain, a useful form of self-adaptation can be cast in terms of context switches it does not explain how the decision to change contexts is arrived at. To understand this aspect of the problem we consider the relationship that $level_n$ has with $level_{n-1}$.

As we have seen above, each level has two relationships with its parent level. First, the child level's program in synthesized by its parent level by (a) using the child level's description as a specification for the child's program, (b) considering the known state of the child's process as repre-

[2] Using a clustering algorithm based upon minimal description length (Principle Component Decomposition – (Robertson, P., Laddaga, R. 2002)

sented in the parent level knowledge-base, and (c) by considering the set of available contexts that were learned for that level. The child level's description is the result of task decomposition of the parent level's description. The child level's relationship to its parent is therefore 'task decomposition'. The parent level also created the child level as the result of its reasoning about contexts, the child specification, and the state of knowledge. That relationship, which consists of the rationale for the decisions about the child's structure, is a meta-relationship. The parent level thus encapsulates four pieces of knowledge about the child: (1) a more abstract representation of the child's structure, (2) knowledge about why the child was constructed the way it was—and what agents were ultimately chosen in its construction, (3) knowledge about the world collected by the child, and (4) how well the child has been doing at its task. Of the above, (1) and (2) relate to the construction of the child before the child had a chance to run, while (3) and (4) relate to knowledge and meta-knowledge, respectively, of the child's execution history. We explain where (4) comes from in the next section.

In this application, we chose to use contexts to drive the child process structure, however, in a different application another approach might have been taken. The use of contexts in this application represents an offline compilation of alternative self-adaptive choices.

Whatever approach is taken, the parent level is *related* to the child level by two forms of knowledge located in the parent level.

1. The child level is a program tailored to implement the specification of the task-decomposed goal of its parent. The child has no apparatus to reason about why it does what it does or how it might do differently if things don't work out well in the current environment—but the parent level does.
2. The parent represents the knowledge that the child is using the agents chosen for it precisely because, at the time it was constructed, those agents were most suitable for performing the tasks decomposed from the parent's specification.

Knowing how well the Child layer is doing

The execution of any level results in an interpretation of the image chosen so as to produce the MDL representation. The description length is, in part, a measure of goodness of the interpretation. The program can produce an interpretation of a blank image as an 'urban scene' because of smoothing[3] but the description length will be large. When a context begins to change, such as when the image in the sequence begins to transition from say a rural landscape to a suburban one, more parts of the image will suffer from being alien to the context—resulting in the use of more smoothed rule and the inevitable growth in description length for the image. Description length of an image interpretation by a level is governed by two factors: (1) the complexity of the image being interpreted, and (2) the goodness of fit of the context used to create the level. Factor (1) tends to be dominated by the ground area covered by the image. When the contexts were established (offline) the defining set of images that formed the cluster had interpretations whose description length was known. From that set of images' description lengths a model of the typical description length of an image is established as a Gaussian. Since the goodness of fit of the context dominates the changes in description length, serious deviations from the mean (outliers) indicate a change in context.

In addition to meta-reasoning, reasoning about the task decomposition also takes place in order to consider the need to rebuild a new child program. Finally, the goal of the program is also subject to change. If the priorities of the system change during a run, the system can self-adapt to reflect those changes. Adaptation therefore is driven by these three aspects of the system state:

1. Meta-knowledge about system performance on the current images.
2. The abstract representation of the knowledge extracted from the image.
3. The changing state of the goals for the system.

We have focused on the first of these mechanisms in this paper. The three sources of knowledge are managed by a theorem prover that plays a central role in the child program synthesis.

Conclusions

We have interleaved the description of the GRAVA architecture and of the satellite aerial image understanding program that was implemented using it. Some of the choices made in the implementation, such as the learning and subsequent use of contexts, were arbitrary and driven by the problem domain, others, such as the relationships between parent and child levels, are general and would be common to other GRAVA applications. The domain discussed in this paper uses an efficient and simple representation of program structure choices (contexts) but more complex domains could use richer representations with correspondingly richer meta-reasoning. We have described how self-adaptation draws upon both meta-knowledge and abstract knowledge in order to effect adaptation to changes in the environment (changing makeup of the scene).

[3] Smoothing is a method whereby rules are created to ensure the success of an operation (such as a grammatical rule). Rules generated by smoothing ensure success but entail a large description length.

While many image interpretation programs will merrily continue to interpret images that don't reflect the programs domain of competence, the described program, by using meta and abstract knowledge, catches itself in the act of doing something silly and self-adapts into a program that can do better. In this way meta-reasoning leads to a program that, through self-knowledge, produces robust performance—and where a good interpretation cannot be arrived at through self-adaptation it can at least indicate that its best interpretation is unreliable.

Acknowledgements

Effort sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Material Command, USAF, under agreement number F30602-98-0056. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

References

Robertson, P. 2002. *A Self-Adaptive Architecture for Image Understanding*. DPhil Thesis, University of Oxford.

Robertson P., Laddaga, R. 2002 *Principle Component Decomposition for Automatic Context Induction,* Proceedings of the Artificial and Computational Intelligence conference, Tokyo Japan

Terras, M., Robertson, P. 2005 *Image and Interpretation. Using Artificial Intelligence to Read Ancient Roman Texts*, Human IT

Terras, M. 2007 *Image to Interpretation—Using Artificial Intelligence to Read the Vindolanda Texts*.
in Image to Interpretation An Intelligent System to Aid Historians in Reading the Vindolanda
Texts (chapters 4 and 5), Oxford University Press.

Robertson, P. Laddaga, R.*2003 GRAVA: An Architecture Supporting Automatic Context Transitions and its Application to Robust Computer Vision,* Proceedings of the 4th International and Interdisciplinary Conference CONTEXT 2003, Stanford, CA

Raja, A. Lessor, V. 2004 ***Meta-Level Reasoning in Deliberative Agents*** Proceedings of the Intelligent Agent Technology, IEEE/WIC/ACM International Conference

Erman, L. D. Hayes-Roth, F. Lesser, V. R. and Reddy, D. R. 1980. ***The HEARSAY-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty.*** *ACM Computing Surveys* 12(2).

Cox, M. T., and Raja, A. 2007. ***Metareasoning: A Manifesto***, Technical Report, BBN TM-2028, BBN Technologies. www.mcox.org/Metareasoning/Manifesto