A High-Level Language for Homeland Security Response Plans

Richard Scherl and **Michael Barnathan**
Computer Science Department
Monmouth University
West Long Branch, NJ 07764
rscherl@monmouth.edu s0510795@monmouth.edu

Abstract

In this paper we discuss the applicability of the agent control language GoLog for the representation of plans to handle the response to homeland security incidents. Our goal is a representation for plans that is high-level and flexible. The specific actions executed when the plan is interpreted should be selected through logical inferencing to be the appropriate actions under current circumstances. Our approach is illustrated with an example of agricultural bio-terrorism.

Introduction and Motivation

After a homeland security incident has occurred (no matter of what type), the speed of the response is of utmost importance. The amount of detail that human analysts must take into account is immense. We propose the specification of high-level plans to be executed automatically by agents/programs to relieve some of the responsibilities from human analysts. The hope is that the execution of these plans which incorporate both common-sense and specialized background knowledge will exhibit some of the flexibility of human responders.

Our goal is a language that allows the specification of relatively abstract high-level plans. When the plans are evaluated (i.e. interpreted by an agent) the specific actions to be executed are selected through logical inferencing given the background knowledge. For example, the response to an act of bio-terrorism can be specified in general terms, but the specifics depend upon the particular biological agent suspected and the specifics of the location.

First we briefly describe the GoLog language and its variants. Our plan specification language is being built upon GoLog. Next an illustrative bio-terrorism example is described. Finally, current and future work is discussed.

GoLog

GoLog(Levesque *et al.* 1997; Reiter 2001) is a high-level agent programming language built on the situation calculus. The idea behind GoLog is to integrate reasoning, perception, and action within a uniform theoretical and implementation framework.

GoLog programs are constructed out of a number of high-level constructs representing complex actions. These are then evaluated by the interpreter through logical inferencing and the result is a series of primitive actions which can be physically executed. The evalution occurs in the context of a background logical theory. The following constructs for complex actions are available in basic GoLog:

- $\delta_1; \delta_2$ – sequences
- $\delta_1 | \delta_2$ nondeterministic choce of actions
- **if** ϕ **then** δ_1 **else**δ_2 – conditionals
- **while** ϕ **do**δ – while loops
- $(\Pi x)\, \delta$ – nondeterministic choice of parameters.
- recursive procedures.

There are currently a number of variants of GoLog available and Prolog interpreters have been developed for many of these. Basic GoLog has been augmented with concurrent and reactive features in the variant ConGolog(Giacomo, Lespérance, & Levesque 2000). Additionally, knowledge and sensing have been added to basic GoLog (Scherl & Levesque 2003; Reiter 2001). IndiGolog (Giacomo & Levesque 1999; Lespérance & Ng 2000) combines the concurrent and reactive features of ConGolog with sensing, but a restricted version of knowledge.

Ignoring the differences between the variants, the result is a high-level language for the specification of plans. The details are filled in at execution time through logical inferencing. The language is suitable for writing *knowledge-based programs* in which some of the actions to be performed are sensing actions needed to obtain the necessary knowledge to determine which course(s) of action must be carried out.

Agricultural Bio-Terrorism Example

To illustrate our approach, we utilize the Silent Prairie agricultural bio-terrorism exercise(Zdenka 2004; Strategic Policy Forum 2004; Parker 2002) developed by the National Strategic Gaming Center at the National Defense University. The exercise/scenario begins with cases of suspected food and mouth disease in North Carolina and Kansas. It should be noted that although foot and mouth disease does not affect humans, the introduction of the disease would have a devastating effect on the agricultural economy of the United States.

Among the many actions that need to be taken initially are:

- Notify the FBI since terrorism is suspected.
- Notify state officials and farm companies.
- Determine to which states N. Carolina cattle are shipped and notify the governors.
- Send sample to Plum Island facility for analysis.
- Start initial containment strategy (quarantine zones).
- Notify USDA, FDA, DHS, and DOD.
- Enact regional containment strategy.
- Check to see if there was "chatter" about a FMD act of terrorism being planned.

Further actions depend upon the results of these initial actions.

Here is a piece of the GoLog like plan. The plan specifies a module that determines to which states animal f (in our case cattle) are sent from a (in our case North Carolina) and performs the appropriate notification:

proc det_notify(a)
While $\neg$**knows**$(\forall x \text{State}(x) \rightarrow \text{Considered}(x))$
$(\Pi x).\text{state}(x)?;$
If $\neg$**Kwhether**$(\text{Shipto}(f, a, x))$ **then**
sense_ship(f, a, x) **endif**;
If Knows$(\text{Shipto}(f, a, x))$ **then**
notify(f, a, x) **endif**;
endWhile;
endProc

Note that each state is considered in turn. The sensing action which needs to consult databases of shipments is not executed if it is already known whether or not shipments are made to that state. Otherwise, the sensing action is executed and then if it is known that cattle are shipped to that state, the notification is executed.

As a further example, consider the gathering of evidence that the incident is a result of criminal/terrorist activity. This demands the ability to access information from a variety of sources. These include records of "chatter", databases on the biowarfare/bioterrorism interests/capabilities of different countries, and sources with information on biological agents. Integrating and fusing the data available from the different sources (with different terminologies, schemas) demands reasoning with background knowledge about the nature of the information provided by each source.

Current and Future Work

Currently, an implementation of the above scenario is being built within the agent platform Jade (`http://jade.tilab.com`). The controlling agent executes the GoLog plan for responding to the potential incident of agricultural Bio-Terrorism. In doing so, the agent requests information from various other agents (written in Java) and sends commands to be "executed" by yet other Java agents.

Future work will include.

- Development of natural-language facilities to make the construction of GoLog programs possible for those who do not know the details of the language.
- Integration of reasoning about information annotated in semantic web ontology languages such as OWL.
- Adding a probabilistic representation of belief, potentially inaccurate information, and probabilistic effects of actions.

Summary

In conclusion, the work so far demonstrates that GoLog is a promising basis on which to develop a high-level language for the representation of plans suitable for responses to homeland security incidents. Further experimentation with different scenarios is needed to determine which features need to be present in a suitable language for this application.

Acknowledgments

This work has been supported by the Knowledge Fusion Center of the Army Research Laboratory under contract number DAAD-03-2-0034. We thank Yves Lesperance for helpful conversations on GoLog and its variants.

References

Giacomo, G. D., and Levesque, H. J. 1999. An incremental interpreter for high-level programs with sensing. In *Logical Foundations for Cognitive Agents: Contributions in honor of Ray Reiter*. Springer-Verlag. 86–102.

Giacomo, G. D.; Lespérance, Y.; and Levesque, H. J. 2000. ConGolog, a concurrent programming language based on the situation calculus. *Artificial Intelligence* 109–169.

Lespérance, Y., and Ng, H. 2000. Integrating planning into reactive high-level robot programs. In *Proceedings of the Second International Cognitive Robotics Workshop*, 49–54.

Levesque, H.; Reiter, R.; Lespérance, Y.; Lin, F.; and Scherl, R. B. 1997. Golog: A logic programming language for dynamic domains. *Journal of Logic Programming*.

McCarthy, J. 1968. Programs with common sense. In Minsky, M., ed., *Semantic Information Processing*. The MIT Press. chapter 7, 403–418.

Parker, H. 2002. *Agricultural Bioterrorism: A Federal Strategy to Meet the Threat*. Washington, D.C.: Institute for National Strategic Studies.

Reiter, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. Cambridge, Massachusetts: The MIT Press.

Scherl, R., and Levesque, H. 2003. Knowledge, action, and the frame problem. *Artificial Intelligence* 144:1–39.

Strategic Policy Forum. 2004. Silent prairie participant guide. unpublished exercise materials prepared by the National Defense University.

Zdenka, W. 2004. Silent prairie crisis simulation. Presentation on April 14, 2004 at Monmouth Univerisity using materials prepared by the National Defense University.