Verification of Emergent Behaviors in Swarm-based Systems

Christopher Rouff, Amy Vanderbilt
SAIC
rouffc@saic.com

Mike Hinchey
NASA GSFC Code 581
michael.g.hinchey@nasa.gov

Walt Truszkowski, James Rash
NASA GSFC Code 588
walter.f.truszkowski@nasa.gov

Abstract

The emergent properties of swarms make swarm-based missions powerful, but at the same time more difficult to design and to assure that the proper behaviors will emerge. We are currently investigating formal methods and techniques for verification and validation of swarm-based missions. The Autonomous Nano-Technology Swarm (ANTS) mission is being used as an example and case study for swarm-based missions to experiment and test current formal methods with intelligent swarms. Using the ANTS mission, we have evaluated multiple formal methods to determine their effectiveness in modeling and assuring swarm behavior. This paper introduces how intelligent swarm technology is being proposed for NASA missions, and gives the results of a comparison of several formal methods and approaches for specifying intelligent swarm-based systems and their effectiveness for predicting emergent behavior.

1. Introduction

A significant challenge when verifying and validating swarms of intelligent interacting agents is how to determine that the possible exponential interactions and emergent behaviors are producing the desired results.

We have investigated formal methods and techniques for verification and validation of swarm-based missions. The advantage of using formal methods is their ability to mathematically assure the behavior of a swarm, emergent or otherwise. The Autonomous Nano-Technology Swarm (ANTS) mission is used as an example and case study for swarm-based missions for which to experiment and test current formal methods with intelligent swarms. We have evaluated multiple formal methods to determine their effectiveness in modeling and assuring swarm behavior.

This paper introduces how intelligent swarm technology is being proposed for NASA missions, and give the results of a comparison of several formal methods and approaches for specifying intelligent swarm-based systems. Example specifications are given to illustrate the advantages and disadvantages of each method.

2. Swarm Technology and ANTS Overview

Bonabeau et al. [2] who has studied self-organization in social insects stated "that complex collective behaviors may emerge from interactions among individuals that exhibit simple behaviors" and described emergent behavior as "a set of dynamical mechanisms whereby structures appear at the global level of a system from interactions among its lower-level components." These emergent behaviors are the sums of simple individual behaviors, but when aggregated together form complex and often unexpected behaviors.

Intelligent swarms [1] are based on swarm technology where the individual members of the swarm have independent intelligence. This makes verifying such systems even more difficult since the swarms are no longer made up of homogeneous members with limited functionality and communications.

In addition to emergent behavior in swarms, there are also a large number of concurrent interactions going on between the agents that make up the swarms. These interactions can contain errors, such as race conditions, that are difficult to detect until they occur. Once they occur, it can be difficult to recreate the errors since they are usually data and time dependent.

With intelligent swarms, members of the swarm may be heterogeneous or homogeneous. Further, homogeneous swarms, due to their differing environments, may learn different things, develop different goals and therefore become a heterogeneous swarm. Intelligent swarms may also reflect different capabilities as well as a possible social structure. This creates a huge state space. With learning, the behavior of individual elements and the emergent behavior, the swarm will be constantly changing and its behavior will be difficult to predict for testing purposes.

The Autonomous Nano-Technology Swarm (ANTS) mission [3] will have swarms of autonomous satellites that will search the asteroid belt for asteroids with specific characteristics. There will be approximately 1,000 spacecraft involved in the mission. To implement this mission a high degree of autonomy is being planned that will be near total autonomy. A heuristic approach is being considered that provides for a social structure to the spacecraft based on a hierarchy. Crucial to the mission will be the ability to modify its operations autonomously to reflect the changing nature of the mission and the distance and low bandwidth communications back to Earth.

There will be several types of spacecraft involved in the mission (Figure 1). Some of the spacecraft will be Leaders that have rules and goals that decided the types of asteroids and data the mission is interested in and will coordinate the efforts of the workers. The third type of spacecraft are messengers and will coordinate communications between the workers, leaders and Earth. Leaders contain models of the types of science they want to perform. Parts of this model are communicated to the messenger spacecraft that then relay it on to the worker spacecraft. Teams would work together to form models of asteroids as well as form virtual instruments.

3. Approaches and Assurance

As mission software becomes increasingly more complex, it also becomes more difficult to test and find errors. This is especially true of highly parallel processes and distributed computing, such as swarms. Race conditions in these systems can rarely be found by inputting sample data and checking if the results are correct. These types of errors are time-based and only occur when processes send or receive data at particular times or in a particular sequence or after learning occurs. To find these errors, the software processes involved have to be executed in all possible combinations of states (state space) that the processes could collectively be in. Because the state space is exponential to the number of states, it becomes untestable with a relatively small number of processes. Traditionally, to get around the state explosion problem, testers have artificially reduced the number of states of the system and approximated the underlying software using models.

Formal methods are proven approaches for assuring the correct operation of complex interacting systems [4, 10, 11]. They are particularly useful for specifying complex parallel and distributed systems where more than one person was involved in the development. Once written, a formal specification can be used to prove properties of a system correct, check for particular types of errors (e.g. race conditions), as well as used as input to a model checker. Verifying emergent behavior is one area that most formal methods have not addressed.

We surveyed formal methods techniques to determine if there existed formal methods that would be suitable for verifying swarm-based systems and their emergent

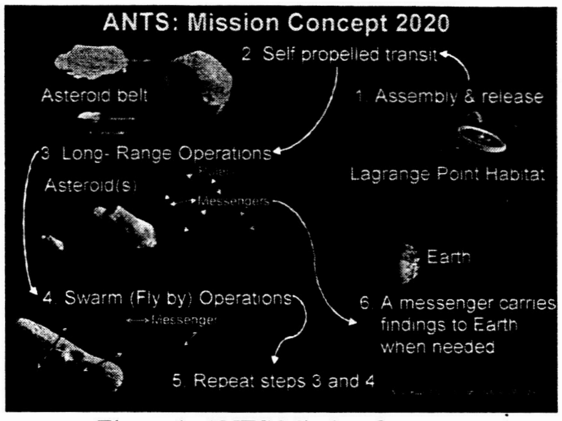

Figure 1: ANTS Mission Concept.

behavior. It was found that there are a number of formal methods that support either the specification of concurrency or algorithms [12]. Though there were a few formal methods that have been used to specify swarm-based systems, only two formal approaches had been found that were used to analyze the emergent behavior of swarms. Weighted Synchronous Calculus of Communicating Systems (WSCCS), a process algebra, was used by Tofts to model social insects [14], and to analyze the non-linear aspects of social insects [13]. X-Machines have been used to model cell biology [8, 9] and modifications have potential for specifying swarms. Simulation approaches are being investigated to determine emergent behavior. These approaches do not predict emergent behavior from the model but model the emergent behavior only after the fact.

4. Specifications and Evaluation

As described in the initial evaluation of specification techniques for swarm-based systems, specifications of the NASA ANTS mission was done using Communicating Sequential Processes (CSP), Weighted Synchronous Calculus of Communicating Systems (WSCCS), Unity Logic and X-Machines. Here we provide partial specifications of ANTS using the four methods, an evaluation of these methods and their potential for analyzing emergent behavior. In each case, only enough of the ANTS mission was specified to gather enough information to evaluate the method for specifying swarm-based systems. The remainder of this report gives the evaluation of the above methods.

4.1. CSP

Each of the spacecraft has goals to fulfill their mission. The emergent behavior of all these goals should equal the goals of the mission. The following is the top-level specification of the ANTS mission:

$$ANTS_{goals} = Leader_{i,l\_goals} \parallel Messenger_{j,m\_goals} \parallel Worker_{k,w\_goals} \bullet 1 \le i \le m,\ 1 \le j \le n,\ 1 \le k \le p$$

where m is the number of leader spacecraft, n the number of messenger spacecraft and p the number of worker spacecraft. The ANTS mission starts, or is initialized, with a set of goals given to it by the principal investigator and part of these goals are given to the leader (some of these goals may not be given to the leader because the goals are ground based or not applicable to the leader). The leader spacecraft specification consists of two processes:

$$Leader_i = LEADER\_COM_{i,\{\}} \parallel LEADER\_INTELLIGENCE_{i,goals,model}$$

the communications process and the intelligence process. The communication process, *LEADER_COM*, specifies the behavior of the spacecraft as it relates to communicating with the other spacecraft and Earth, and specifies a protocol between the spacecraft. The second process, *LEADER_INTELLIGENCE*, is the specification of the intelligence of the leader. This is where the deliberative and reactive parts of the intelligence are implemented and the maintenance of the goals for the leader is done. In addition to the goals, the *LEADER_INTELLIGENCE* process also maintains the models of the spacecraft and its environment and specifies how it is modified during operations. Each of the above processes has parameters that have an identifying number that indicates which spacecraft of a group it is, as well as other parameters that are sets that store conversations, goals and models. Since at startup there have been no conversations, the conversation set in the LEADER_COM process is empty. Since leaders are given initial goals and models, these sets are non-empty at start up. The following is an example portion of a top level specification of the leader communication:

$$\begin{aligned}
&LEADER\_COM_{i,conv} = leader.in?msg \rightarrow \\
&\text{case } LEADER\_MESSAGE_{i,conv,msg} \text{ if sender}(msg) = LEADER \\
&MESSENGER\_MESSAGE_{i,conv,msg} \text{ if} \\
&\text{sender}(msg) = MESSENGER, WORKER\_MESSAGE_{i,conv,msg} \\
&\text{if sender}(msg) = WORKER, EARTH\_MESSAGE_{i,conv,msg} \\
&\text{if sender}(msg) = EARTH, ERROR\_MESSAGE_{i,conv,msg} \\
&otherwise
\end{aligned}$$

4.2. WSCCS

To model the ANTS Leader spacecraft, WSCCS (Weighted Synchronous Calculus of Communicating Systems), a process algebra, takes into account:

- The possible states (agents) of the Leader
- Actions each agent-state may perform that would qualify them to be in those states
- The relative frequency and priority of each action

Agent states and view of priority (p) and frequency (f) on the actions of the Leader as seen in Table 1.

Based on this, the states of the Leader can now be

Table 1: Leader States and Actions

State	Action	f	[illegible]
	Identity		
Commun-icating	SendMessageWorker	50	[illegible]
	SendMessageLeader	50	[illegible]
	SendMessageError	1	
	ReceiveMessageWorker	50	[illegible]
	ReceiveMessageLeader	50	[illegible]
	ReceiveMessageError	1	
Reasoning	ReasoningDeliberatve	50	[illegible]
	ReasoningReactive	50	[illegible]
Processing	ProcessingSortingAndStorage	17	[illegible]
	ProcessingGeneration	17	[illegible]
	ProcessingPrediction	17	[illegible]
	ProcessingDiagnosis	16	[illegible]
	ProcessingRecovery	16	[illegible]
	ProcessingRemediation	17	[illegible]

defined by definition statements such as the following:

$$\begin{aligned}
&Communicating \equiv \\
&50\omega^2 : \text{ReasoningDeliberatve}.Reasoning + \\
&50\omega^2 : \text{ReasoningReactive}.Reasoning \\
&+ 17\omega^2 : \text{ProcessingSortingAndStorage}.Processing \\
&+ 17\omega^2 : \text{ProcessingGeneration}.Processing \\
&+ 17\omega^2 : \text{ProcessingPrediction}.Processing \\
&+ 16\omega^2 : \text{ProcessingDiagnosis}.Processing \\
&+ 16\omega^2 : \text{ProcessingRecovery}.Processing \\
&+ 17\omega^2 : \text{ProcessingRemediation}.Processing
\end{aligned}$$

This statement is saying that Leader, when in a Communicating state, has the option (is allowed) to perform any action from the set

$$\{\text{ReasoningDeliberatve}, \text{ReasoningReactive}, \text{ProcessingSortingAndStorage}, \text{ProcessingGeneration}, \text{ProcessingPrediction}, \text{ProcessingDiagnosis}, \text{ProcessingRecovery}, \text{ProcessingRemediation}\}$$

and that the Communicating Leader will perform ReasoningDeliberatve with a probability of 25% and will give that action the same priority as the others. The second term in the statements tells us that the Communicating Leader will perform ReasoningReactive with the same 25% probability and priority of 2. The symbol + in this notation denotes that the Communicating Leader will make a choice between the various allowed actions, and that that choice will be made based on the frequencies and priorities of each allowable action.

The single Leader by itself shows the following example emergent behavior. The Communicating Leader Will choose to transition to a Processing state with a probability of 50% by choosing to process by one of the sic available processing types. It will choose from the six types with equal probability.

To study the emergent behavior of a swarm of Leaders we begin by considering a swarm of only 2 Leader spacecraft; call them L1 and L2. Then both leaders tick forward by performing one action per time step. Thus the two Leaders perform a composition of two actions, denoted $m1\omega^{k1} * m2\omega^{k2}$, on each time step. When this happens, the pair of leaders behaves according to the rules for composition:

[illegible]

This gives the Leader pair their own set of relative frequencies and priorities. Since there are two Leaders and each has three states and 14 possible actions, the pair of leaders has 9 possible state pairs and 196 possible action compositions.

The 2-Leader swarm will have a much higher probability of having both leaders communicating or reasoning, rather than processing. Processing will be done by the swarm, but with much less frequency than communicating or reasoning. These features can be extrapolated to the swarm of n leaders as follows.

Given a swarm of n Leader Spacecraft, the n-leader swarm will tick forward in time by performing simultaneous actions – one action per leader per time step. Thus the n-leader swarm will perform (on each time step) a composition of n actions, denoted with weight $m_1\omega^{k_1} * m_2\omega^{k_2} * \ldots * m_n\omega^{k_n}$. When this happens, the n-leader swarm still must behave according to the rules for composition seen before.

This gives the n-leader swarm its own set of relative frequencies and priorities. Since there are n Leaders and each has three states and 14 possible actions, the swarm of n leaders has 3^n possible state sets and 14^n possible action compositions. There are only two possible priority values and four possible relative frequency values available and thus we can narrow down that each priority k_i must be either 1 or 2 and each relative frequency m_i must be either 1 (if the priority is 1) or one of 16, 17 or 50 (if the priority is 2). Any composition which includes any leader communicating in error will have a priority less than the priority of not sending any messages in error and thus the swarm will not choose to send or receive a message in error. Thus the remaining options for leaders in the swarm will include communicating (not in error), reasoning, and processing (either by prediction or recovery, or otherwise). Let N_{comm} be the number of leaders in the swarm who choose to communicate (not in error) on a given time step. Let N_{reason} be the number of leaders in the swarm who choose to reason on that time step. Let $N_{process16}$ be the number of leaders in the swarm who choose to process (by prediction or recovery) on that time step. Lastly, let $N_{process17}$ be the number of leaders in the swarm who choose to process (by other means) on that time step.

Then, each action by each leader will have priority 2 and relative frequency 16, 17 or 50. Thus, the composition of their actions will have weight

$$m_1\omega^{k_1} * m_2\omega^{k_2} * \ldots * m_n\omega^{k_n} = (50^{N_{comm}+N_{reason}})(16^{N_{process16}})(17^{N_{process17}})\omega^{2n}$$

From this weighting, we can see that drastically higher frequencies exist when larger numbers of the leaders in the swarm choose to communicate or reason. Much lower frequencies exist when larger numbers of leaders choose to process. Thus the swarm will be communicating and reasoning much more often than processing, although processing will take place.

4.3. Unity Logic

To model the ANTS Leader spacecraft with Unity Logic, we consider states of the Leader just as in WSCCS and other state - machine based specification languages. In Unity Logic, we will consider the states of the Leader, and the actions taken to make the Leader be in those states, but the notation will appear much closer to classical logic. Predicates will be defined to represent the actions that would put the Leader into its various states. Those predicates then become statements which, if true, would mean that the Leader had performed an action that put itself into the corresponding state. The Leader program would then be specified using ***assertions*** such as the following for Communication:

[Communicating]ReasoningDeliberatve(Leader)[Reasoning]
[Communicating]ProcessingGeneration(Leader)[Processing]

Unity Logic then provides a logical syntax equivalent to Propositional Logic for reasoning about these predicates and the states they imply as well as for defining specific mathematical, statistical and other simple calculations to be performed.

4.4. X-Machines

To model the ANTS Leader spacecraft as an X-Machine we must be able to see the Leader as a tuple
$L = \{Input\ , Memory\ , Output\ , Q\ , \Phi\ , F\ , start\ , m_0\}$
where the components of the tuple are defined as

$$Input = \begin{Bmatrix} worker, messenger, leader, error, \\ Deliberative, Reactive, \\ SortAndStore, \\ Generate, Predict, Diagnose, \\ Recover, Remediate \end{Bmatrix}$$

$Memory$ will be written as a tuple $m = (Goals, Model)$ where Goals describes the goals of the mission and Model describes the model of the universe maintained by the Leader. The initial memory will be denoted by $(Goals_0, Model_0)$. When the goals and/or model changes, the new tuple will be denoted as $m' = (Goals', Model')$.

$$Output = \begin{Bmatrix} SentMessageWorker, \\ SentMessageMessenger\ , SentMessageLeader\ , \\ SentMessageError\ , ReceivedMessageWorker, \\ ReceivedMessageMessenger, \\ ReceivedMessageLeader\ , \\ ReceivedMessageError\ , \\ ReasonedDelibartively\ , ReasonedReactively\ , \\ ProcessedSortingAndStoring, \\ ProcessedGeneration\ , ProcessedPrediction\ , \\ ProcessedDiagnosis\ , ProcessedRecovery, \\ ProcessedRemediation \end{Bmatrix}$$

$Q = \begin{Bmatrix} Start, Communicating, \\ Reasoning, Processing \end{Bmatrix}$ is a set of states.

$\Phi = \begin{Bmatrix} SendMessage, ReceiveMessage, \\ Reason, Process \end{Bmatrix}$ is a set of (partial) transition functions where each transition function maps $Memory \times Input \rightarrow Output \times Memory$ as in the following:

$$\Phi(m, Worker) = (m', SentMessageWorker)$$
$$\Phi(m, Generate) = (m', ProcessedGeneration)$$

Then $F : Q \times \Phi \rightarrow Q$ is defined according to definitions such as in Table 2

Table 2: Leader States and Transitions

Q	Φ	$Q' = F(Q,\Phi)$
Start	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing
Commun-icating	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing
Reasoning	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing
Processing	SendMessage	Commun.
	ReceiveMessage	Commun.
	Reason	Reasoning
	Process	Processing

5. Evaluation of Methods

CSP is a process algebra and is very good at specifying the process protocols between and within the spacecraft and analyzing the result for race conditions. Being able to evaluate a system for race conditions is very important in highly parallel systems. From a CSP specification, reasoning about the specification can be done to determine race conditions as well as converted into a model checking language for running on a model checker.

WSCCS provides a process algebra that takes into account the priorities and probabilities of actions performed by the leader and other ANTS spacecraft. It further provides a syntax and set of rules for predicting and specifying choices and behaviors, as well as a congruence and syntax for determining if two automata are equivalent. All of this in hand, WSCCS can be used to specify the ANTS spacecraft and to reason about and even predict the behavior of one or more spacecraft. This robustness affords WSCCS the greatest potential for specifying emergent behavior in the ANTS swarm. What it lacks towards that end is an ability to track the goals and model of the ANTS mission in a memory. This may be achieved by blending the WSCCS methods with the memory aspects of X-Machines.

Unity Logic provides a logical syntax equivalent to simple Propositional Logic for reasoning about predicates and the states they imply as well as for defining specific mathematical, statistical and other simple calculations to be performed. However, it does not appear to be rich enough to allow ease of specification and validation of more abstract concepts such as mission goals. This same simplicity, however, may make it a good tool for specifying and validating the actual Reasoning programming (as opposed to Reasoning process) portion of the ANTS Leader spacecraft, when the need arises. In short, specifying emergent behavior in the ANTS swarm will not be accomplished well using Unity Logic.

X-Machines provide a highly executable environment for specifying the ANTS spacecraft. It allows for a memory to be kept and it allows for transitions between states to be seen as functions involving inputs and outputs. This allows us to track the actions of the ANTS spacecraft as well as write to memory any aspect of the goals and model. This ability makes X-Machines highly effective for tracking and affecting changes in the goals and model. However, X-Machines do not provide any robust means for reasoning about or predicting behaviors of one or more spacecraft, beyond standard propositional logic. This will make specifying emergent behavior difficult.

A blending of the above methods seems to be the best approach for specifying swarm-based systems. Blending the memory and transition function aspects of X-Machines with the priority and probability aspects of WSCCS may produce a specification method that will allow all the necessary aspects for specifying and predicting emergent behavior in the ANTS mission and other swarm-based systems.

7. References

[1] Beni, G. and Want, J. Swarm Intelligence. In Proceedings of the Seventh Annual Meeting of the Robotics Society of Japan, pp 425-428, Tokyo, Japan, 1989, RSJ Press.

[2] Bonabeau, E., G. Theraulaz, et al. Self-organization in Social Insects, Trends in Ecology and Evolution, 1997, vol. 12, pp. 188-193.

[3] Curtis, S. A., J. Mica, J. Nuth, G. Marr, M. Rilee, and M. Bhat. ANTS (Autonomous Nano-Technology Swarm): An Artificial Intelligence Approach to Asteroid Belt Resource Exploration. International Astronautical Federation, 51st Congress, October 2000.

[4] Hinchey, M. Jarvis, S. Concurrent Systems: Formal Development in CSP. McGraw-Hill. 1995.

[5] Hinchey, M. and Bowen, J. Industrial-Strength Formal Methods in Practice. Springer. 1999.

[6] Hoare, C.A.R. Communicating Sequential Processes. Communications of the ACM, 21(8):666-677, August, 1978.

[7] Holcombe, M. Towards a formal description of intracellular biochemical organization. Technical Report CS-86-1. 1986. Dept of Computer Science, Sheffield University, United Kingdom.

[8] Holcombe, M. Mathematical models of cell biochemistry. Technical Report CS-86-4. 1986. Dept of Computer Science, Sheffield University, United Kingdom.

[9] Holzmann, H. J, Design and Validation of Computer Protocols, Prentice Hall Software Series, Englewood Cliffs, NJ, 1991.

[10] Nayak, P. Pandurang, et. al. 1999. Validating the DS1 Remote Agent Experiment. In Proceedings of the 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS-99).

[11] Rouff, C., Rash, J., Hinchey, M. Experience Using Formal Methods for Specifying a Multi-Agent System. Sixth IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2000) September 11-15, 2000.

[12] Rouff, C., Vanderbilt, A., Truszkowski, W., Rash, J., and Hinchey, M. Formal Approaches to Intelligent Swarm Technology, A Survey of Formal Methods for Intelligent Swarms. Technical Report. 2003. http://ants.gsfc.nasa.gov.

[13] Sumpter, D.J.T., Blanchard, G.B. and Broomhead, D.S. Ants and Agents: A Process Algebra Approach to Modelling Ant Colony Behaviour. Bulletin of Mathematical Bilogy. 2001, 63, 951-980.

[14] Tofts, C. Describing social insect behaviour using process algebra. Transactions on Social Computing Simulation. 1991. 227-283.