Mechanism of Failure Diagnosis based on Language Games and Q-learning over a Termites Simulator

Arles Rodríguez Portela*
Alife Research Group
National University of Colombia
aerodriguezp@unal.edu.co

Jonatan Gómez Perdomo†
Alife Research Group
National University of Colombia
jgomezpe@unal.edu.co

ABSTRACT

This paper presents a mechanism of failure diagnosis in a multi-agent environment of termites looking for food. The failure system is defined based on the probability that each termite (agent) has of executing a movement instruction. By using language games concepts and a version of the Q-learning algorithm, termites diagnose failures with the highest failure probabilities. Termites also have enough information to determine which movement actuator is failing based on a simple voting system that is the result of language games of diagnosis. Results show that the proposed approach is able, from local interactions, to build a set of very specific diagnosis questions, allowing the system to diagnose more than one type of failure at the same time, while the accounted number of diagnosis questions for instructions with low failure probability is reduced.

Categories and Subject Descriptors

I.2.6 [**Computing Methodologies**]: Artificial Intelligence-learning

General Terms

Algorithms, Design.

Keywords

Multiagent environment, termites simulator, language games, Q-learning, failure diagnosis.

1. INTRODUCTION

Self-healing is based on the ability to detect software and hardware components that are failing. System must detect failures in components and then, replace, eliminate or repair them without disrupting the system operation. Self-healing involves: the design and verification of an autonomic system which has some of the complexity of a real system in order to locate functions and services offered by an autonomic element in an efficient manner [6], to make an abstraction of behaviors to obtain emergent properties and global behaviors from local actions [6, 2, 3], to reallocate resources [1] and to locate faults in the shortest possible time[5].

In this paper, a simulator of a termite's swarm, a failure system, and a diagnosis mechanism are introduced. Termites are modeled as agents with a virtual machine that execute instructions about motion and diagnosis. Ant Colony System (ACS) is used to locate a point of food in the space. A simple failures system, in which each action has a probability of error (termites have no knowledge about this probability) is designed. Agents also have the possibility to diagnose others; we use language games and Q-learning to provide the diagnosis mechanism.

2. THE DIAGNOSIS MECHANISM

Termites are modeled as agents that look for food, carry the food to home and continue searching for more using Ant Colony System (ACS)[4]. Each agent executes a simple program that encapsulates ACS and the mechanism of diagnosis that executes Language Games. A program example is exposed in Algorithm 1. `SEEK` and `CARRY` are simple instructions that execute ACS and `DIAGNOSE` starts the diagnosis mechanism when a termite has another termite as neighbor.

Algorithm 1 Program example for termites

```
1: IFSEEKING
2: SEEK
3: IFCARRYING
4: CARRY
5: IFHASNEIGHBORS 1
6: DIAGNOSE
7: GOTO 0
```

A failure is defined as a probability associated with each actuator of movement by each termite. When a termite executes a movement instruction, a random number in the interval [0,1] is generated. If the random number is greater than the failure probability, the instruction is executed. For example, a termite with a failure vector as in Table 1 cannot execute the "Left" movement. Termites have no knowledge about this failure vector.

The `DIAGNOSE` instruction is based on language games. A basic naming language game [7] is modified into the following steps (Diagnosis instructions are defined on Table 2.):

*A. Rodríguez is a Bachelor in Computer Science of Universidad Distrital Francisco José de Caldas. Currently, he is coursing a Master in Computer Science at National University of Colombia.

†J. Gómez is Ph. D. in Mathematics with concentration in Computer Science of Memphis University (August 2004). Currently, he works as Associate Professor at National University of Colombia.

GECCO'11, July 12–16, 2011, Dublin, Ireland.
ACM 978-1-4503-0690-4/11/07.

Table 1: Failure vector for a termite

Action	Down	Left	Right	Up
Error Prob	0	1	0	0

- **Making contact:** Two agents are physically close and make contact one to another. One assumes the role of speaker and the other will be the hearer.
- **Diagnose:** The speaker chooses one action from its vector of diagnosis (using Q-learning) and the hearer is informed about it using the `ACT` instruction.
- **Action:** The hearer does the action, in this case it makes a movement in the world, and reports the result of the action to the speaker using the `DONE` instruction.
- **Feedback:** The speaker computes the result of the action and compares its result with the final position that it was expecting. If the result is the same, the diagnosis ends in failure (it does not discover a possible failure), the hearer gets a negative vote and the question of diagnosis is punished. Otherwise, the hearer receives a positive vote and the diagnose question is rewarded.

Table 2: Diagnose instructions

Instruction (syntax)	Definition
ACT *(action,posX,posY)*	Indicates to the hearer to do an action *action* and the speaker location.
DONE *(result,posX, PosY)*	Indicates to the speaker that action is done, its result and the hearer location. If result is true the action was realized.
DIAGNOSE (result-action-posX,posY)	Indicates to hearer the result of a diagnosis for the action *action* and hearer location.

Termites have a vector called belief vector of failures that stores the voting of all the neighbors that have done a diagnosis to a termite. If an error is found in the diagnosis process, a value of 1 is added for this action, otherwise a value of -1 is subtracted (no failure found). Table 3, shows a voting of 20 for an error in the Left movement, and a negative vote for the Up movement.

Table 3: belief vector of failures for a termite

Action	Down	Left	Right	Up
Error Prob	0	20	0	1

A vector of diagnosis stores the questions of diagnosis about actions and weights associated with each action. Q-learning is used to optimize the questions of diagnosis. Each question of diagnosis has an initial weight of $1/\#actions$. If the speaker's results do not match with the hearer's, the question of diagnosis about this action receives a reward, otherwise the question receives a punishment. The question of diagnosis with the greatest value is chosen. If more than one question has the same greatest value, the first question is chosen.

3. EXPERIMENTS AND RESULTS

Populations of ten termites with one, two, three and five failures in different actuators and 20 termites with five failures are defined. In all the experiments a failure probability of one (1) is defined for the movement actuators. All the termites have the same failures in their actuators. The vector of questions of diagnosis is compared to the belief vector of failures to get the results.

For ten termites and one failure, the ten termites diagnose the failure. For three failures and ten termites, the three types of failures are diagnosed. For five failures and 10 termites, three of the five types of failure are detected. For 20 termites and five failures, the five failures are detected. In all cases, very few questions of diagnosis are generated for actuators without failures and the votes of the belief vector of failures tends to be zero. For experiments with more than one type of failure, the termites are specialized in a different kinds of failure.

A diagnosis mechanism based on language games and Q-learning is presented. Each termite is able to identify its own failures given the diagnosis of others. Local interactions in the mechanism of diagnosis allow system to be specialized in the detection of more than a failure at the same time. If a failure has a higher probability than other, Q-learning will give a reward to this question of diagnosis, the voting for this action will be increased on the belief vector of failures and questions about actions with the lowest probability will decrease significantly.

4. ACKNOWLEDGMENTS

This work was supported by the DIB (Dirección de investigación sede Bogotá- Universidad Nacional de Colombia) and its program "Apoyo de la DIB a tesis de investigación en posgrados - Segundo Corte" code 12153, and by the alife research group - National University of Colombia.

5. REFERENCES

[1] H. Arora, T. Raghu, A. Vinze, and P. Brittenham. Collaborative self-configuration and learning in autonomic computing systems: Applications to supply chain. pages 303–304, June 2006.

[2] N. Bicocchi and F. Zambonelli. Autonomic communication learns from nature. *Potentials, IEEE*, 26(6):42–46, Nov.-Dec. 2007.

[3] T. De Wolf and T. Holvoet. Towards autonomic computing: agent-based modelling, dynamical systems analysis, and decentralised control. pages 470–479, Aug. 2003.

[4] M. Dorigo and L. M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, 1997.

[5] J. Gao, G. Kar, and P. Kermani. Approaches to building self healing systems using dependency analysis. 1:119–132 Vol.1, April 2004.

[6] J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, Jan 2003.

[7] L. Steels and P. Vogt. Grounding adaptive language games in robotic agents. In *Proceedings of the Fourth European Conference on Artificial Life*, pages 474–482. MIT Press, 1997.