Complexity Analysis of Task Dependencies in an Artificial Hormone System

Eric Hutter, Mathias Pacher and Uwe Brinkschulte

Institut für Informatik, Goethe University, Frankfurt am Main, Germany
{hutter,pacher,brinks}@es.cs.uni-frankfurt.de

We study an extension of the Artificial Hormone System (AHS) [1], a decentralized middleware to assign tasks to the processing elements (PEs) of a distributed system. The AHS is based on Organic Computing Principles and has no single point of failure. To achieve this, the AHS establishes decentralized control loops based on the exchange of short digital messages, so-called *hormones*. In case a PE fails, its tasks are automatically redistributed to healthy PEs. Furthermore, the AHS is real-time capable: It is possible to prove hard time bounds for initial task allocation as well as re-allocation after PE failures.

However, the AHS only considers independent tasks. Thus, we extend it in this paper by introducing new *negator* hormones that allow tasks to inhibit other tasks' execution and assignment. For example, in Figure 1, T_i can only be executed if T_j is not currently executed. This allows to establish complex task relationships in a simple way.

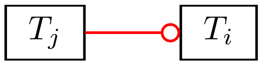

***Figure 1**: Negator relationship between T_j and T_i*

We show that the introduction of negators increases the computational complexity of seemingly simple decision problems. We define the problem NEGATOR-SAT as follows:

Given some task set S, a task $A \in S$ and infinite PE capacities, is there a distribution of the tasks to the available PEs so that

1. *No further task can be assigned,*
2. *No negator relationship is violated,*
3. *Task A is assigned to some PE.*

We show that while the answer is always "yes" for the original AHS,[1] the introduction of negator relationships makes NEGATOR-SAT much harder. In fact, we prove that it is NP-complete by reducing 3-SAT.

For example, the task set $S = \{X_1, X_2, X_3, \overline{X_1}, \overline{X_2}, \overline{X_3}, \overline{C_1}, \overline{C_2}, A\}$ along with the negator relationships shown in Figure 2 corresponds to CNF formula $f = (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_1 \vee x_2 \vee x_3)$: Conditions 1-3 can only be satisfied simultaneously if the assignment state of the literal tasks $X_i/\overline{X_i}$ corresponds to a satisfying interpretation for f.

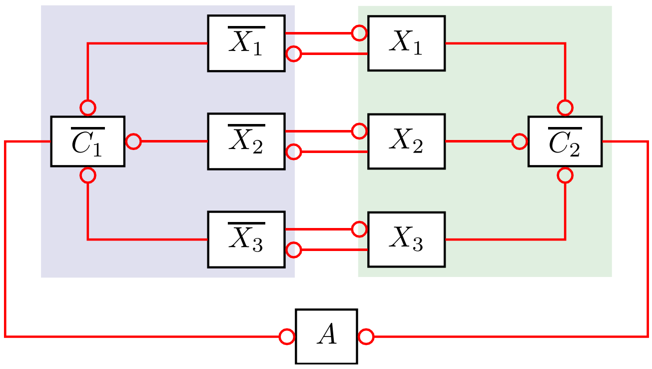

***Figure 2**: NEGATOR-SAT instance constructed from 3-SAT formula*

A short evaluation confirms the theoretical results by showing our construction works in practice. We further discuss the practical relevance of our findings: The introduction of negators enables designers to build much more capable systems using the AHS. But, as even seemingly simple questions like "can a given task be executed *at all*" become much harder to decide algorithmically, great care has to be taken while designing a system employing negator relationships.

References

[1] U. Brinkschulte, M. Pacher, and A. von Renteln, "An Artificial Hormone System for Self-Organizing Real-Time Task Allocation in Organic Middleware," in *Organic Computing*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 261–283.

[1] As long as A can be executed on any PE