

European Symposium on Intelligent Techniques

*March 20-21, 1997
Bari, Italy*

organized by:

European Network in Uncertainty Techniques
Developments for Use in Information Technology

in co-operation with:

Politecnico di Bari
Dipartimento di Elettrotecnica ed Elettronica

supported by the European Commission
DG III Industry - Esprit Programme as a Network of Excellence



Proceedings

A DISTRIBUTED, CONCURRENT ARCHITECTURE FOR FUZZY EVOLUTIONARY MODELS

Ferdinando Cicalese, Antonio Gisolfi and Vincenzo Loia
Dipartimento di Informatica ed Applicazioni
Università di Salerno
84081 Baronissi (Salerno) ITALY

ABSTRACT: This paper proposes the use of actor technology as high level tool to design evolutionary systems. Actory programming allows the achievement of concurrent computing without facing low-level, hardware/software parallel feature. Actor are provided with fuzzy knowledge base which undergoes evolutionary process. Fuzziness and Evolutionary characteristics allow a deeper representation of open information system evolution also enabling to simplify the design of the knowledge base of actors.

1. INTRODUCTION

Genetic algorithms (GA) mimick an evolution process in order to solve an optimization problem. Each individual represent a potential solution and is encoded numerically into a set of chromosomes. A fitness function establishes a value of survival of the individual : the closer the corresponding numerical function to a certain criterion, the better fitness that individual is given.

Artificial evolution genetic algorithms (AEGA) are a particular class of genetic algorithms. Generally AEGA are useful for three kinds of research: natural evolution, evolution of complex behaviour of artificial organisms, function optimisazion.

The major issue in designing evolving systems, is the capability to represent efficiently adaptation [Loia and Scandizzo 1995, 1996]. As pointed out in [Fogel 1996] adaptation consists of three aspects : prediction, control, feedback.

This work proposes the Actor Model as a metaphor of high level evolutionary distributed computations. The Actor Model, initially introduced in [Hewitt 1977] has been computationally defined in [Agha 1986].

Each actor embodies passive information via its own acquaintances - slots containing data - and reacts to external stimuli by triggering its scripts - a type of internal functioning. Actors are individual entities which execute local tasks when external messages are received and stocked in their mail-box. Communication is allowed via a message passing protocol, which is based on a simple concept: the external knowledge of an actor corresponds to a set of addresses of its neighbours. The actors work asynchronously in a parallel universe to achieve common goals by means of collaborative plans; each of these actors possesses partial knowledge of their environment, but potentially, through message passing of tasks, it gets a global perspective of the overall net. We decentralise control by delegating a designed actor to accomplish specific duties and by augmenting the distribution of control activities.

Actors have a deadline to accomplish their tasks and are deemed to expire once this time has come. Their results are defined in terms of a function they have to maximise.

When finished actors spread out their knowledge base and resources, which are used to build up new actors. New actors may be identical to the previous one (if they demonstrated optimal behaviour) ; more probably they would show innovative feature, which eventually enable the survived actor to cope with the new situation.

To treat with simplicity the implementation details of our framework we adopt in this paper the ESAL formalism [Dattolo and Loia 1996] , briefly introduced in the following section.

2. ACTOR MODEL: SEMANTICS FOR OPEN INFORMATION SYSTEMS

Open Information Systems (OIS) are large-scale information systems that react to continuous interactions with the external world. In [Hewitt 1991], a deep reflection about the role of distributed Artificial Intelligence (DAI) in OIS underlines the necessity to provide a framework in which important issues of AI are integrated.

In this paper we face the problem of finding appropriate high level software paradigms able to represent the computational complexity of OIS. Due to the deep interaction, and consequently to the impossibility to maintain a closed world assumption, it is necessary to radically change the usual -algorithmic- approach useful and efficient whenever a unique output is delivered for every input. The irreducibility of algorithms to interactive behaviour [Wegner 1995] forces us to adopt more suitable modelling paradigms to represent *interaction machines*, i.e. extensions of Turing machines towards open systems.

The framework is viewed as an environment in which a continual flow of new information is originated from numerous actors. Actors exploit massive concurrency; they own local data (acquaintances) and perform concurrently functions (scripts) which can change due to an evolution of the whole environment. The decentralisation of knowledge and tasks is not an obstacle to global actions; cooperative, collaborative duties (some of these based on hybrid mechanisms) may be used to coordinate their local objectives. Message-passing facilities is the only communication strategy allowable for the actors.

The actor model satisfies the double requirement of high-level programming and efficiency. Actors combine object-oriented and functional programming in order to make easier for the user the management of concurrency. Briefly, the actor model can be synthesized as follows:

- the universe contains computational agents, called *actors*;
- actors perform computation through asynchronous, point-to-point message passing;
- each actor is defined by its state, mail queue and behaviour;
- an actor's state is defined by its internal data, called *acquaintances*;
- an actor reacts to the external environment by executing its *scripts*.
- scripts are activated according to *fuzzy control rules*, which constitute the *genotype* of an actor.
- a genetic algorithm is used to produce new actors from expiring ones, by processing fuzzy genotypes.

Evolutionary actor programming paradigm represents an interesting approach in treating OIS because it is able to manage concurrent distributed and interactive computations but also to formalise the adaptive and evolutionary transformations affecting an open and competitive environment, where parameters characterising optimal behaviour change from time to time.

In order to make our discussion more abstract, instead of using a specific actor language we prefer to formalise our model via a pseudo-actor language that facilitates the specification of the different actor entities by differentiating task definition and distribution.

2.1 ACTOR DEFINITION

The construct Def is used to define an actor, according to this form:

```
(Def myactor
  with-acquaintances (. . .)
  with-communication-list {task args}
  with-control rules list (. . .)
```

Essentially, an actor is composed of a data part and a script part; this last section is a sequence of possible scripts which can be executed by the actor. Script execution relies on the status of the actor (data) and the messages it receives from other actors.

The causality relationship between actor's activity and actor's status is expressed via *fuzzy control rules*:

$$\begin{array}{l} \text{if } acq_1 \text{ is } \alpha_1, acq_2 \text{ is } \alpha_2, \dots, acq_n \text{ is } \alpha_n \\ \text{then } task_1 \text{ is } \beta_1, task_2 \text{ is } \beta_2, \dots, task_m \text{ is } \beta_m \quad (*) \end{array}$$

where $\alpha_1, \alpha_2, \dots, \alpha_n$, are fuzzy number representing the evaluation of the data variables acq_i ; and $\beta_1, \beta_2, \dots, \beta_m$ are fuzzy numbers representing the causality values of tasks with respect to the given set of data.

TASK ACTIVATION

In the definition of an actor, many fuzzy control rules may appear. The task activated is determined by the following procedure.

Let m be the number of fuzzy rules; the i th fuzzy rule will be referred as:

acq_1 is α_{i1} , acq_2 is α_{i2} , ..., acq_n is α_{in}

then $task_1$ is β_{i1} , $task_2$ is β_{i2} , ..., $task_m$ is β_{im}

$t \leq$

given the status $\{acq_1, acq_2, \dots, acq_n\}$, the task to be activated will be $task_k$ if and only if:

$$\bigcap_{j=1}^n \alpha_{ij}(acq_j) \cap \beta_{ik} = \max_{1 \leq i \leq m} \bigcup_{i=1}^m \left\{ \bigcap_{j=1}^n \alpha_{ij}(acq_j) \cap \beta_{ik} \right\}$$

From a logical point of view, if $\{acq_1, acq_2, \dots, acq_n\}$ is the status, the i th rule says that the j th task can be activated with possibility equal to $\gamma_i = \alpha_{i1}(acq_1) \cap \alpha_{i2}(acq_2) \cap \dots \cap \alpha_{in}(acq_n) \cap \beta_{ij}$. By taking the logical

maximum $\pi_j = \bigcup_{i=1}^m \gamma_i$ we have the overall possibility value for the j th task to be executed, with respect to the set of fuzzy rules.

The task actually activated is the one for which the value π is maximum among the m possible tasks.

ACTOR EVOLUTION

A fuzzy rule (*) can be encoded as a string like the following:

$$c_5 \gamma^5 \dots c_{s-1} \gamma^{s-1} \dots c_1 \gamma^1 (**)$$

where $\{\gamma_1, \gamma_2, \dots, \gamma_k\} = \{\alpha_1, \alpha_2, \dots, \alpha_n\} \cup \{\beta_1, \beta_2, \dots, \beta_m\}$

any c_i is the subset of $\{acq_1, \dots, acq_n\} \cup \{task_1, \dots, task_k\}$, whose elements all have the evaluation γ_i

string (**) is a compact, and value ordered version of the rule (*), where each evaluation appears exactly once.

This new form two fuzzy rules can be aggregated by using the following procedure [Gisolfi & Loia 95]:

Let

$$a_n \alpha_n a_{n-1} \alpha_{n-1} \dots a_1 \alpha_1$$

$$b_m \beta_m b_{m-1} \beta_{m-1} \dots b_1 \beta_1$$

Let

$$B = (a_n \alpha_n a_{n-1} \alpha_{n-1} \dots a_1 \alpha_1) \Delta (b_m \beta_m b_{m-1} \beta_{m-1} \dots b_1 \beta_1) =$$

$$+_{n-1} \gamma^{m+n-1} \dots c_1 \gamma^1$$

where, for $n > m$

$$\gamma_i = \begin{cases} \bigcup_{j=1}^i a_{i-j+1} \cap b_j & 1 \leq i < m \\ \bigcap_{j=1}^m a_{i-j+1} \cap b_j & m \leq i < n \\ \bigcup_{j=i-n+1}^i a_{i-j+1} \cap b_j & n \leq i < m+n \end{cases}$$

where the \cap and \cup are the well known operations of set-join and set-meet;

from rules

acq_1 is α , acq_2 is β then $task_1$ is γ , $task_2$ is α

acq_1 is β , acq_2 is α then $task_1$ is β , $task_2$ is γ

get the representation (supposing $\gamma < \beta < \alpha$):

$$= [acq_1, task_2]^\alpha [acq_2]^\beta [task_1]^\gamma$$

$$= [acq_2]^\alpha [acq_1, task_1]^\beta [task_2]^\gamma$$

and aggregating R_1 and R_2 we obtain a string like

$$= [acq_2, acq_1]^{s_1} [task_2]^{s_2} [task_1]^{s_3}$$

which represents the rule

acq_1 is δ_1 , acq_2 is δ_1 then $task_1$ is δ_3 , $task_2$ is δ_2

where δ_1 , δ_2 and δ_3 , are fuzzy numbers yielded by the fuzzy aggregation operators. This aggregation is the basis for the evolution of the actors' knowledge base. We refer to R_3 as the *offspring* of the mating rules R_1 and R_2 . Given a population of *actors* $Act_1, Act_2, \dots, Act_p$, each actor keeps on with its fuzzy rules for a certain amount of time. During this period it would gain credit as the value of the function it has to maximize (actor's target or fitting function). Actors with large credit can continue their life. Actor with small credit finish to die and be replaced by new actors, whose fuzzy rules are obtained by mating the knowledge base of the well performing actors. Moreover the resulting new population of actors undergoes a mutation operator. Mutation means that actors fuzzy rules can be modified (with small probability) by changing the fuzzy numbers attached to the tasks and acquaintances.

CONCLUSION

The evolution of knowledge representation models from declarative-based approach, in the 1960s, to agent-based approach in the 1990s has increased the interest of research community in how agent-based systems can present and handle knowledge-based decision making processes. This work presents an actor-based model to describe fuzzy evolutionary systems. Different features makes this approach novel and interesting: the actor model is more suitable to represent systems with high interaction [Wegner95] the actor approach introduces concurrency in fuzzy information management; the uncertainty information contained in actors makes more flexible and robust the overall inference methodology of the system; the evolutionary target of the architecture increases the reactivity of the system. This last aspect is more important. "Standard" actor-based models may suffer of staticity (i.e. it is not easy the possibility to transform entirely the actor entities). As future works we intend to investigate about real size case-studies, such as representation and simulation of global markets [Merz and Lamersdorf 1996] [Gasser 1994] and distributed diagnostic systems [Gisolfi and Loia 1996].

BIBLIOGRAPHY

- Ghosh, G. 1986. *Actors: A Model of Concurrent Computation in Distributed Systems*. Cambridge, MA, USA.
- Mattolo, A., Loia, V., Collaborative Version Control in Agent-based Hypertext Environment, *Information Systems*, 21 2 :127-145, 1996.
- Lewitt, C. Viewing control structure as pattern of passing message, *Artificial Intelligence*, 8 pp.326-364, 1977.
- Lewitt, C. Open Information Systems Semantics for Distributed Artificial Intelligence. *Artificial Intelligence*, 7, pp.79-106, 1991.
- Fogel D. B., *Evolutionary Computation*, IEEE Press, 1996.
- Gasser, L. Agent organizations for information retrieval and electronic commerce: the next frontier. Proc. of the Workshop on Heterogeneous Cooperative Knowledge-Base, International Symposium FGCS'94, Japan, December 15-16, pp.49-63, 1994.
- Gisolfi A., Loia V., A Complete Flexible Fuzzy-Based Approach to the Classification Problem, *Int. Journal of Approximate Reasoning*, Vol.13, 15- 183, 1995.
- Gisolfi A., Loia V., Distributed Multiple Diagnosis via Actor Computing, to appear in *Int. J. of Information Sciences*.
- Loia, V., Scandizzo, S. Qualitative Selection Strategies in Genetic-based Evolutionary Economic Models, Proc. of the Joint Third International Symposium on Uncertainty Modeling and Analysis and Annual Conference of the North American Fuzzy Information Processing Society}, Maryland, USA, 17e20 September 1995, pp. 333-338, IEEE Press.
- Loia, V., Scandizzo, S., Networks of Strategic Alliances in a FuzzE Evolutionary Environment. to appear in Proc. of the 4th International Conference on Soft Computing, Iizuka, Japan, September 30 - October 5, 1996.
- Merz, W. Lamersdorf. Agents, Services, and Electronic Markets: How do they Integrate? Proc. of IFIP/IEEE International Conference on Distributed Platforms, Dresden, 1996.
- Wegner P. Models and Paradigms of Interaction, TR-CS-95-21, Dept. Computer Science, Brown University, USA, September 1995.