A TOPOLOGICALLY-DRIVEN STRATEGY TO PREVENT INSTABILITY IN RULE-BASED AUTONOMOUS AGENTS

Victor Zamudio, Vic Callaghan

University of Essex
Wivenhoe Park CO4 3SQ United Kingdom
email:{vmzamu,vic}@essex.ac.uk

Keywords: Pervasive Computing, Instability, Periodic Behaviour, Autonomous Agents, Social Networks, Socio-Economic Systems.

Abstract

Pervasive computing has been found to have a fundamental problem of instability. This problem is due to interaction between distributed but coordinating rule based autonomous agents in pervasive and intelligent environments. It is impossible to predict when a set of rules will lead the system to an unstable state, as it depends on the rules and on the state of the pervasive environment. Clearly, such instability could be a major obstacle to the commercial success of pervasive computing. In this paper we present a method of describing and reasoning about such behaviour, an ***Interaction Network (IN*)**, together with a strategy based on finding the loops and locking the nodes in a way that minimizes the impact on the other devices and the user. This strategy is tested successfully with different topologies and rules.

1 Introduction

Pervasive computing is growing rapidly, with cheaper and better interconnected devices. These interconnections enable the system to be programmed with interdependent actions in a simple way, whether it be manual or automatic [1, 2, 3]. Thus, devices such as lights, heaters, TVs, telephones, etc. could be programmed to perform a task according to certain rules, based on the behaviour of other devices. These rules could be very complex, not only because of the rules but because of the number of devices involved (some which may be nomadic), resulting in unwanted disruptive behaviour. Besides that, temporal delays (due to network latency, speeds of processing, etc.) could result in some devices receiving old information, contributing to unstable behaviour. This phenomenon is being observed increasingly in pervasive computing system as the architectures move from centralized to distributed control [4].

From complex system theory it has been shown that it is not possible to predict theoretically whether an arbitrary set of rules will suffer from instability [5]. However, it is possible to prevent it and we have developed and tested such a strategy, based on the detection of loops in an Interaction Network introduced in the next section, and a method for locking devices with least functional impact on the performance of the system.

2 Interaction Networks – An Introduction

A ***directed graph*** G consists of a finite set V of vertices or nodes, and a binary relation E on V. The graph G is denoted as (V, E). The relation is called the adjacency relation. If w is relative of v (ie, $(v, w) \in E$) then w is adjacent to v [6]. An agent A is an autonomous device with a binary state $s \in \{0,1\}$, where 0 means that the agent is *off*, and 1 means that the agent is *on*. If we have n autonomous devices agents $A_1, A_2, \ldots A_n$ the state of the system is $S = (s_1 s_2 \ldots s_n)$. Each agent A_i has two rules: i) if ϕ_i then $s_i = 1$ ii) If ψ_i then $s_i = 0$ where ϕ_i and ψ_i are boolean functions that depend on the states of then agents.

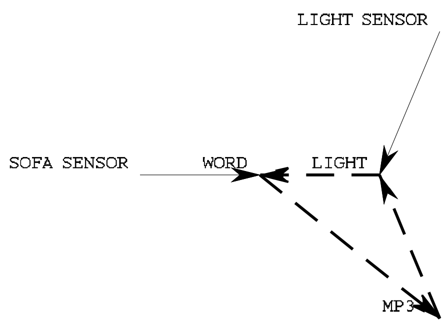

Fig. 1: Interaction Network showing a loop in dashed lines

An ***Interaction Network (IN)*** is a directed graph (V, E) in which the vertex $v \in V$ is a pervasive autonomous agent A and $(v_i, v_j) \in E$ if the Boolean functions ϕ_j or ψ_j of the pervasive autonomous agent A_j depends on the binary state s_i of the agent A_i. Let $U \subseteq S$ be a subset of S. Because of the dynamics of the system, the system will produce a sequence of states $U_1, U_2, \ldots U_p$. If this sequence of states is periodic then the subsystem U is said to be *periodic*.

The ***functionality of a node*** is defined as the number of descendants in the Interaction Network. This characteristic of a node is very important, as it shows the impact of a device in the system, in terms of the number of devices whose rules could be triggered.

Fig. 1 provides an example of an Interaction Network, showing the dependencies of 5 devices or services: Sofa Sensor, Light Sensor, MP3 Player, Light, and Word.

2.1 The Intelligent Locking Strategy

As we mentioned before, it is not possible to predict, given a set of rules and a set of initial conditions, whether a system will show instability (oscillations), rather we identify the potential for instability and lock a device or service to prevent it occurring. Our mechanism[1] for achieving this may be summarised as follows:

- Detect loops in the Interactive Network associated to the system.
- For each loop:
- Find the nodes member of the loop which minimizes the functionality function.
- Lock these nodes (this includes learning the users "locking preferences" where there are choices that affect the user).

3 Implementation and Results

In order to evaluate our approach, we developed a simulator in **Mathematica™ 5.1** [7] Mathematica™ is a powerful and widely used programming language for work that requires abstract processing, visualization and numerical tools. In particular, it includes a package called ***Combinatorica***, which includes tools to support graph theory based work [8].

The simulator has been written so it can automatically generate a random topology and rules of interaction, and run a number of trials showing graphically the dynamics of the system, (using decimal representation of the binary global state of the system). Besides, it is possible to include random perturbations, which can be interpreted as the interaction of the user with the system; which can lead the system to an unwanted periodic behaviour. A mechanism of automatic locking is implemented (see Section 2.1). As the functionality of all the members of a loop has been found to be the same, we excluded the descendant members of the loop in the calculations.

Using different topologies for the Interaction Network, we tested our approach in the following cases (see section 3.1):

1. Acyclic system
2. System with only one loop
3. System with two isolated loops
4. System with two coupled loops

In terms of the dependencies of the rules of interaction, it is possible to have dependencies on 1 variable, or dependencies on more than 1 variable. However, as the main focus of our research relates to the loops (as they provide the feedback of the system) our tests were based on the topological properties of the IN shown in Fig. 2.

$$\text{Topology} \begin{cases} \text{acyclic} \\ \text{cyclic} \begin{cases} \text{isolated} \\ \text{coupled} \begin{cases} 1\text{point} \\ +1\text{points} \end{cases} \end{cases} \end{cases}$$

$$\text{Dependencies} \begin{cases} 1\text{variable} \\ +1\text{variable} \end{cases}$$

Fig 2: Taxonomy of the problem, showing the two approaches: topological and functional.

3.1 Acyclic System

We tested our approach with a system with 7 nodes, where the rules of interaction are encoded as a binary string, with 0 and 1 meaning OR and AND respectively. In this particular case, the rules are {0,1,0,0,1,0,1} meaning that the node 1 is a node OR, node 1 is a node AND, node 3 is a node OR, etc. If the node only depends on the state of a single node, it will mirror the state of that node. The locking vector is a binary string. If the node 5 is locked, there will be a 0 in the 5th position of the vector; otherwise it will be a 1. In this case there is no loop therefore the locking vector is {1,1,1,1,1,1,1}. The topology of the system is given by a set of ordered pairs. For example {1,7} means that there will be an arrow going from 1 to 7 in the Interaction Network. Table 1 summarizes this information.

The initial state of the system is random, and the probability of a perturbation is 0.05.

In Fig. 3 we can see the Interaction Network (left) and the response of the system (right). As we expected, there were no oscillations (as there are no loops).

[1] Patent No: GB 0624827.2.

Number of nodes		7
Topology		{{1,7},{2,6},{3,2},{3,7},{4,1},{4,3}, {4,5},{5,2}, {5,3}, {7,6}}
Number of Cycles		0
Cycles		{}
Coupled		No
Rules of Interaction		{0,1,0,0,1,0,1}
Locking vector		{1,1,1,1,1,1,1}
Oscillations	*Before*	No
	After	No

Table 1: Interaction Network with no cycles

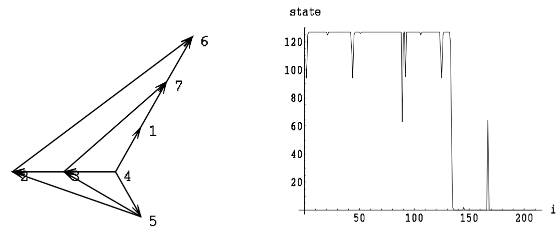

Fig. 3:Interaction Network with no cycles (left). Behaviour of the system showing no instability (right).

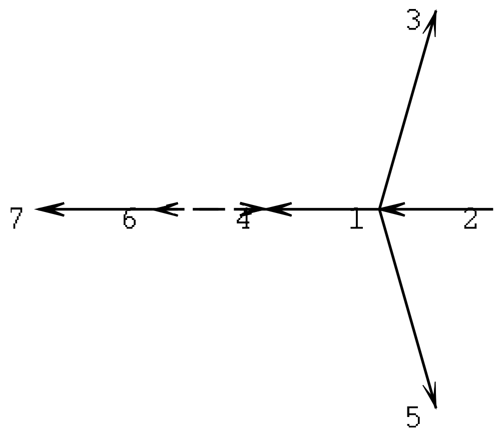

Fig 4: Interaction Network with only one cycle {6,4,6}. Detail of the cycle in dashed lines.

3.2 One Cycle

For this case, we had a system with 7 nodes, and the only loop being {6, 4, 6} (See Fig. 4) Node 6 has 1 descendant, and node 4 had no descendants (as we do not include members of the loop). Node 4 has the minimum functionality (as it doesn't affect anyone outside the loop) and, as a result, the locking vector is {1,1,1,0,1,1,1}(see Table 2). We ran the system several times (with and without locking), and found that the locking mechanism always prevented any instability.

Fig. 5 shows the response of the system without any locking (left) and with the locking (right). Notice that the effect of locking produces a stable system.

Number of nodes		7
Topology		{{1,3},{1,4},{1,5},{2,1},{4,6},{6,4}, {6,7}}
Number of Cycles		1
Cycles		{{6,4,6}}
Coupled		No
Rules of Interaction		{0,0,0,1,0,1,0}
Locking vector		{1,1,1,0,1,1,1}
Oscillations	*Before*	yes
	After	No

Table 2: Interaction Network with only one cycle {6,4,6}

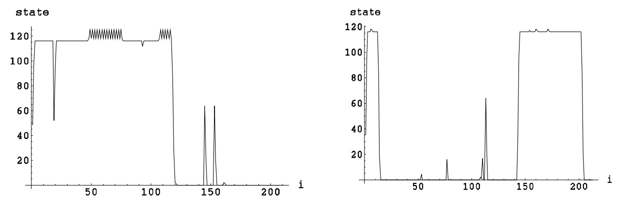

Fig. 5: Evolution of the system with 7 agents, one isolated cycle {6,4,6}. In the left we can see oscillations, and after the locking (right) the oscillations have been removed.

3.3 Two non-coupled cycles

For the case of two uncoupled loops, we used a topology with 12 nodes (these are randomly configured by the simulator). Fig. 6 shows two loops (in dashed lines) {11, 2, 8, 11} and {7, 5, 12, 7}. In the first loop, the node with minimum functionality is 2, and in the second loop all the members have the same functionality, and 7 is locked. The locking vector is {1,0,1,1,1,1,0,1,1,1,1,1}. Table 3 summarizes this information.

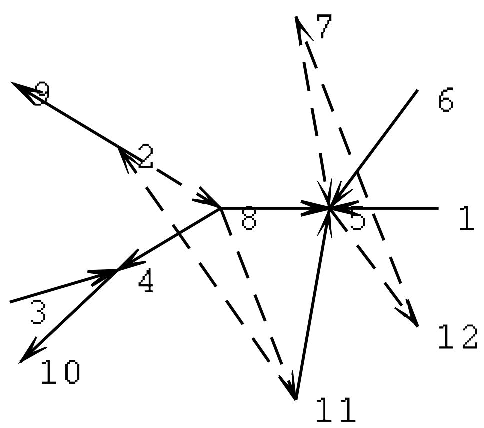

Fig 6: Interaction Network with two uncoupled cycles {11,2,8,11} and {7,5,12,7}. The loops are in dashed lines.

Number of nodes		12
Topology		{{1, 5}, {2, 8}, {2, 9}, {3, 4}, {4, 10}, {5, 12}, {6, 5}, {7, 5}, {8, 4}, {8, 5}, {8, 11}, {11, 2}, {11, 5}, {12, 7}}
Number of Cycles		2
Cycles		{11,2,8,11}, {7,5,12,7}
Coupled		No
Rules of Interaction		{1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0}
Locking vector		{1,0,1,1,1,1,0,1,1,1,1,1}
Oscillations	*Before*	Yes
	After	No

Table 3: Interaction Network with two uncoupled cycles and 12 nodes.

Without locking the system displays three modes of oscillation (see Fig. 7 left). However, by locking nodes 2 and 7 this unstable behaviour is prevented (see Fig. 7 right).

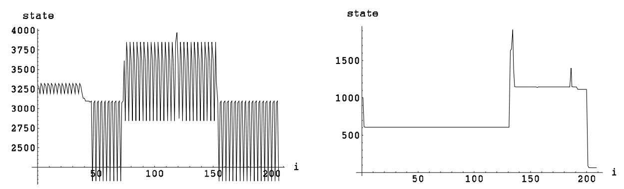

Fig. 7: Evolution of the system with 12 agents and two uncoupled cycles. The system is unstable (left) without locking. When the locking is used the system is stable (right).

3.4 Two coupled cycles

We used the IN shown in Fig. 8 to test the approach for the case of coupled system. We have the two loops {8,5,6,7,8} and {4,2,3,5,4}, sharing node 5. The list of parents-descendants for the loops are {{4, 1}, {2, 1}, {3, 2}, {5, 4}} and {{8, 0}, {5, 6}, {6, 1}, {7, 0}} and the two nodes to be locked are 4 and 8, defining the locking vector {1, 1, 1, 0, 1, 1, 1, 0, 1, 1}. Table 4 summarizes this information.

Number of nodes		10
Topology		{{2,1}, {2,3}, {3,1}, {3,5}, {3,10}, {4,2}, {4,9}, {5,4}, {5,6}, {6,7}, {6,10}, {7,8}, {8,5}}
Number of Cycles		2
Cycles		{{8,5,6,7,8}, {4,2,3,5,4}}
Coupled		yes
Rules of Interaction		{0, 0, 0, 0, 1, 1, 1, 1, 1, 0}.
Locking vector		{1, 1, 1, 0, 1, 1, 1, 0, 1, 1}.
Oscillations	*Before*	Yes
	After	No

Table 4:Interaction Network with 10 nodes and two coupled cycles

As in previous experiments, we ran the simulation several times, first without any locking, and then with locking. The locking mechanism always removed the unstable behaviour (see Fig. 9).

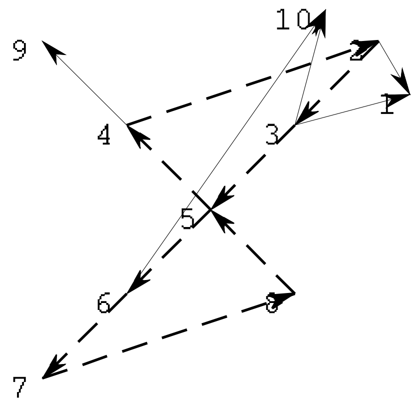

Fig.8: Interaction Network with 10 nodes and two coupled cycles. In dashed lines we have the two loops, sharing node 5.

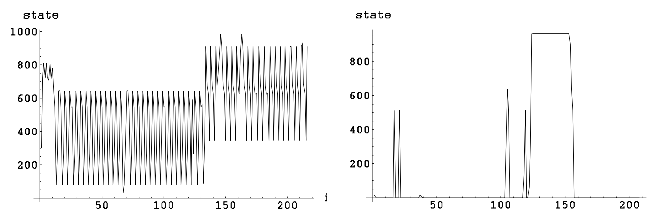

Fig. 9: Evolution of the system with 10 agents and two coupled cycles. The system is unstable (left) without locking. When the locking mechanism is activated, the system is stable (right).

3.5 Results Discussion

We have implemented and tested the strategy of locking with 4 different topologies: an acyclic system, a single loop, and two loop structures (coupled and isolated). When the system is acyclic, there are no instabilities; however, the presence of loops could take the system to an unwanted periodic behaviour with oscillations. Once a loop is found, the node which minimizes the functionality loss (ie the one that has fewer descendants), it is locked. In terms of the services provided to the user, this is very important, as we want to minimize the impact on the user satisfaction.

The random perturbations (playing the role of a user interacting with the environment) are important, as they can lead the system from one mode of oscillation to another (as can be seen in Fig. 7 and 9). In terms of the rules of interaction, every device (or node) is a Boolean node, as they behave as a boolean gate AND (when a 1 is assigned) or as a Boolean gate OR (when a 0 is assigned).

This simplification of the rules allows us to test different topologies, the rationale behind this being that loops are the most significant aspect of the problem (along with the initial conditions and the user interaction).

From the above it is clear that locking has the potential to an effective means to prevent instability, and in that respect we find these results sufficiently encouraging to take this research to the next stage, discussed in the following section.

4 Diversification of IN Applications

4.1 What is an Agent?

So far, in this paper, there has been the implicit assumption that an agent is a software or hardware process. However, the inspiration for the creation of agents is the desire to mimic some of the capabilities that people have; especially those relating to reasoning, planning and learning [13]. Thus, it seems reasonable to describe people as agents, and societies as multi-agent systems. Further, in law, a company is deemed to be equivalent to a person. Companies exist in a complex world where they function autonomously; reasoning, planning and learning about their environment based on interactions with customers, suppliers and competitors. A similar argument can be made for governments and their global behaviours. For the purposes of this research, we view agents in very broad terms being variously machines, people, companies, government and, in fact, any organisation that operates autonomously basing its behaviour on the actions of others, in some way that would be equivalent to a rule based system (eg explicit rules & procedures or implicit habits or behaviours). This model of the world has led us to explore the diversification of the application of Interaction networks in the following ways.

4.2 Interaction Networks: A Generic View

In the last years, the area of social networks has been shown to provide, together with multi-agent systems, useful tools to analyse and represent our world as a complex socio-technical system [9]. Economies, culture, companies and societies can be seen as distributed autonomous systems, with complex time-dependant rules and dynamic interconnections [12]. Work has been done in this direction, in particular trying to analyze and destabilize terrorist networks, finding and removing the leaders of such organizations [9]. In this domain, the presence of loops in the network could suggest redundant leadership and therefore a robust system; our strategy offers a way to analyze and reason about this problem, exposing redundant leaders in a given organization.

Economic behaviour, where entities involved could either have very well defined rules or just try to mimic the behaviour of other participants [10] could show instable behaviour under proper conditions. For example, in share trading, business strategy and global enterprises the behaviour of others is frequently a key factor. Our work offers a tool to explain, analyze and (when appropriate) suppress any unwanted cyclic behaviour.

Finally, research in knowledge networks (who-knows-what), information networks (what ideas are related to what), assignment networks (who is doing what) and social networks will continue gaining relevance in our complex world [9]. In particular, the research on “small worlds” [11] which captures properties from biological and social systems, will continue growing in the years to come.

In all these applications Interaction Networks are a valuable tool where systems involve complex interactions between entities, as they can expose cyclic instabilities due to the rules. Such rules can be either explicit such as written procedures people follow, or implicit rules such as established behaviours or habits that grow from experience arising from embedding of people or organisations in an environment.

In this section we have sought to make an analogy between the behaviour of groups of embedded-agent and people or organisations. Our next step will be to modify our simulation to see if Interaction Network Theory can be used to reproduce some of the better know cyclic behaviours found in social and organisation based systems.

5 Conclusions and Future Work

5.1 Summary

In this paper we have described how coordinating multi agent systems are susceptible to instability (a cyclic behaviour). As pervasive computing paradigms, such as ambient intelligence, utilise systems of interdependent agents, such behaviour represents a serious obstacle to the commercial success of this technology. To counter this problem we have proposed and described a methodology for describing (Interaction networks), identifying (closed loop searching) and elimination (locking nodes) of this behaviour. We have produced a taxonomy that categories the instability problem, in terms of topological and functional issues
We have presented results involving a number of differing topologies that show that these methods are effective in preventing instabilities in pervasive computing systems..

5.2 Further Work

Having proved the underlying principles our next step is to test this strategy with more complex topologies (in particular with multiple coupled loops), and with more complex rules. Also, as locking a node will impair some functionality of the system, the choice of what to lock and how long to lock (where there are options) is of some significance to the user. Thus a next step in our work is to experiment with a user based “locking preference” system learning For this we plan to run experiments in our test bed (iDorm[2] – a full size apartment that is fitted with pervasive computing technology and agents) in order to provide physical evidence of the strategy, and to refine the locking mechanism with information of the user’s preferences.

Finally, whilst this paper has focused principally on interactions between networked embedded-agents, we have made an analogy between embedded agents and people and organisations. From a literature review, we have found that cyclic behaviour of the type we are reporting in this paper can be found in other types of distributed autonomous interacting system, such as financial systems and government. Based on our analogy between agents and other social systems we have proposed that interaction networks could be used to understand and modify cyclic behaviour in these systems. Clearly, this is a challenging direction for our research. However,, based on our work to-date, we believe there is considerable potential for this line of research and we look forward to reporting on our progresses in these areas in future papers.

Acknowledgements

Victor Zamudio would like to acknowledge the support of the National Mexican Council for Science and Technology (CONACyT).

References

[1] V. Callaghan, M. Colley, H. Hagras, J. Chin, F. Doctor, G. Clark. “Programming iSpaces: A Tale of Two Paradigms”, in iSpaces. Springer Verlag, 2005, Chapter 24.

[2] J. Chin, V. Callaghan, G. Clarke. “An End-User Programming Paradigm for Pervasive Computing Applications”, International Conference on Pervasive Services, 26-29 June 2006, Lyon, France.

[3] H. Hagras, V. Callaghan, M. Colley, G.S. Clarke, A. Pounds-Cornish, H. Duman, 'A Fuzzy Logic Embedded-Agent Approach to Ambient Intelligence in Pervasive Computing Environments', IEEE on Intelligent Systems, 2004.

[4] D. Estrin, D. Culler, K. Pister, G. Sukhatme. Connecting the physical world with pervasive networks. Pervasive Computing, IEEE. Jan-March 2002, Volume: 1, Issue: 1, pages: 59-69.

[5] G. Weisbuch. Complex Systems. Lecture Notes Volume II. Santa Fe Institute Studies In the Sciences of Complexity. 1991.

[6] G. Haggard, J. Schlipf and S. Whitesides. Discrete Mathematics for Computer Science. Thomson 2006.

[7] S. Wolfram. The Mathematica Book, 5th ed. Wolfram Media, 2003.

[8] Pemmaraju S, Skiena S. Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica™. Cambridge University Press 2003.

[9] K. M. Carley, J. S. Lee, D. Krackhardt. Destabilizing Networks. Connections 24(3);79-92. 2002.

[10] R.L. Axtell. Effects of Interaction Topology and Activation Regime in Several Multi-Agent Systems. Brooking Working paper, http://brook.edu/ES/dynamics/papers/interaction.

[11] D.J. Watts and S.H Strogatz. Collective Dynamics of Small-World Networks. Nature, 393: 440-442. 1998.

[12] Clarke G, Callaghan V, "Ubiquitous Computing, Informatization, Urban Structures and Density”, Built Environment Journal, Vol. 33, No 2 2007

[13] Callaghan V, Colley M, Clarke G, Hagras H, "A Soft-Computing based Distributed Artificial Intelligence Architecture for Intelligent Buildings", In the book entitled “Soft Computing agents: New Trends for Designing Autonomous Systems”, in the International Series "Studies in Fuzziness and Soft Computing“, (Eds: V. Loia, S.Sessa), Springer Verlag, Volume 75, pp. 117-145, 2002

[2] http://iieg.essex.ac.uk/idorm