Controlling Chaos in Distributed Systems*

Tad Hogg and Bernardo A. Huberman

Abstract

We describe a simple and robust procedure for freezing out chaotic behavior in systems composed of interacting agents making decisions based on imperfect and delayed information. It is based on a reward mechanism whereby the relative number of computational agents following effective strategies is increased at the expense of the others. This procedure, which generates a diverse population out of an essentially homogeneous one, is able to control chaos through a series of dynamical bifurcations into a stable fixed point. Stability boundaries are computed and the minimal amount of diversity required in the system is established.

1 Introduction

Distributed artificial intelligence (DAI) [3] is characterized by the interaction of many agents trying to solve a variety of problems in cooperative fashion. In these systems, each agent has an individual, local view of the problem, and at the same time must successfully interact with other agents in order to find global solutions to the problems posed. One of the difficulties in both the design and the understanding of distributed artificial intelligence systems comes from the lack of central controls, and the ensuing conflicting, uncertain, incomplete and delayed knowledge on the part of the agents. While considerable effort is being devoted to understanding the detailed interactions among a few agents and designing operational DAI systems that can deal with simple problems, relatively little is known about the global behavior of these systems as they are scaled up to deal with more realistic problems. This is an important issue since the individual agents must not only be designed to cooperate with each other, but also adapt appropriately to global consequences of their actions.

Large systems offer a wide range of choices and strategies for addressing different aspects of the overall problem. Thus a major question is how agents should choose among problem-solving strategies and, in particular, with which other agents they should collaborate. Since the appropriateness of these choices can change as agents receive new information and update their states, the system acquires global dynamical characteristics which can work for or against the successful completion of the task. For example, if the time evolution of the agent interactions is simple and therefore predictable, it is relatively easy to program the agents to cope with this variation.

However, when computational agents in these systems make choices in the presence of delayed and imperfect knowledge about the state of the system, their dynamics can become extremely complex, giving rise to nonlinear oscillations and chaos [6, 7, 2]. Part of the complexity in behavior is due to the coevolving nature of the system, i.e., the value of an agent's decision depends on the choices made by the all the others. This complicated dynamics can lead many agents to make poor decisions when judged from a global perspective even though they appear locally reasonable. This leads to significantly lowered performance, and great difficulty in programming the individual agents. Thus, it is important to devise effective decentralized control methods that can simplify the global dynamics. While there are existing procedures for controlling chaotic systems with very few degrees of freedom [5, 10], these are inapplicable to distributed AI systems characterized by a very large number of interacting autonomous agents.

The problem of locally controlling a distributed system can be addressed in a number of ways. First, one could deliberately increase the uncertainty in the agents' evaluation of the merits of different choices and thereby stabilize the system [7]. In this case however, performance degrades enormously because agents tend to ignore actual differences among the choices. A second possibility entails a very slow decision making rate

*appeared in *IEEE Trans. on Systems, Man and Cybernetics* **21**, 1325–1332 (1991).

on the part of the agents, which, once again, is known to produce stability. Unfortunately, in this case the agents aren't responsive to changes in the system, making it nonadaptive. Another method is to increase the diversity of the system by introducing additional types of agents which use different problem-solving methods, since heterogeneous systems tend to be more stable than homogeneous ones. Such a mechanism however, does not provide a way to cope with unexpected changes in the nature of the system such as new task requirements. Thus, more sophisticated approaches are needed to avoid chaos while maintaining high performance and adaptability to unforeseen changes.

A fairly involved approach, analogous to the technical analysis of market behavior, relies on agents' observing patterns in the system's behavior and using these to predict future behavior. Unfortunately, the unpredictability of chaos implies that extrapolating from observed patterns is unlikely to give effective predictions in the chaotic case. Moreover, the extreme sensitivity of chaotic systems to perturbations means that any changes introduced by an agent based on forecasts of the state of the system are likely to change the actual behavior. In particular, when many agents use this technique the dynamics of the system changes markedly, due to the use of a local procedure that assumes that the remainder of the population is not using predictive estimates.

At the opposite extreme, game theoretic techniques can work well if all the agents adopt them and are aware of the exact requirements of the other agents, but fail if these stringent requirements aren't met. Recent studies have shown that in some cases a hybrid of game theory and technical analysis can avoid these difficulties [8]. In this approach, the agents adapt or learn by comparing their predictions with what actually happens. More generally, this can be viewed as providing a mechanism whereby agents eventually come to use those techniques which provide higher payoffs.

Another approach, and the one taken in this paper, is to adopt a policy that requires little knowledge on the part of the agents about the state of the system. This policy is based on agents having a variety of techniques for evaluating choices, and a reward mechanism that increases the proportion of those that are doing well. As we show theoretically and in a number of computer experiments, this simple procedure leads to a robust control of the global performance of the system.

An interesting consequence of this novel way of controlling chaos in distributed systems is the appearance of an interplay between two aspects of coevolution which by themselves cannot stabilize the system: 1) a mechanism for increasing the relative number of those agents using good strategies, and 2) a fluctuating internal environment which continuously changes the fitness of the available strategies. The first aspect by itself leads to a rapid loss of diversity with the consequent inability to control the system, whereas the second in isolation generates the chaotic behavior one is trying to control. In unanticipated fashion, the interaction between these two mechanisms can generate the needed diversity to keep the system stable.

This paper is organized as follows. In §2 we present the basic model and show the consequent dynamics in the absence of a control mechanism. A reward mechanism is introduced in §3 and its effectiveness in controlling chaos and nonlinear oscillations in distributed systems is evaluated in §4. The value of the diversity generated in the system is analyzed in §5, and in §6 we discuss the implications of the procedure and its generalizations.

2 Interacting Agents as an Ecosystem

To evaluate the global dynamics of distributed systems, and the consequences of control methods, we consider a simple model that captures some of the key features of DAI systems. Specifically, a useful view of distributed artificial intelligence systems, in which agents make local choices with imperfect information in order to accomplish their tasks, considers them as instances of *computational ecosystems* [6] by analogy with similar biological and human organizations. The agents' choices can be among strategies or other agents with which to interact. From the point of view of the individual agent, this amounts to the selection of appropriate resources for the solution of the problem. Moreover, since decisions aren't centrally controlled, the agents independently and asynchronously select from among the available choices the one with the highest perceived payoff. These payoffs reflect the actual benefit accrued from the various choices, such as the time required to complete a task, accuracy of the solution, etc. Because the results of various choices can depend on what the other agents do, these payoffs will typically depend on the choices made by other agents.

For simplicity in analyzing the global behavior of large systems we take the payoff G_r for using resource

r to depend on the *number* of agents already using it rather than exactly which agents these are. Although simple, this dependence captures some general properties. For instance, in a purely competitive environment, the payoff for using a particular resource tends to decrease as more agents make use of it. As one possible example, consider a particular heuristic technique: while it may be beneficial for some agents to try it, too many making the same selection could result in wasted effort and, more importantly, missed opportunities to solve the problem with other methods. Alternatively, the agents using a resource could assist one another in their computations. For instance, agents using a particular database could leave index links that are useful to others making the same selection. In such cooperative situations, the payoff of a resource would then increase as more agents use it. In general, we expect some combination of these cooperative and competitive effects.

Imperfect information about the state of the system causes each agent's perceived payoff to differ from the actual value, with the difference increasing when there is more uncertainty in the information available to the agents. This type of uncertainty concisely captures the effect of many sources of errors such as some program bugs, heuristics incorrectly evaluating choices, errors in communication and mistakes in interpreting sensory data. Specifically, the perceived payoffs are taken to be normally distributed, with standard deviation σ, around their correct values. In addition, information delays cause each agent's knowledge of the state of the system to be somewhat out of date. Although for simplicity we will consider the case in which all agents have the same effective delay, uncertainty, and preferences for resource use, we should mention that the same range of behaviors is also found in more general situation [7].

As a specific example, we consider the case of two resources, so the system can be described by the fraction f of agents that are using resource 1 at any given time. Its dynamics is then governed by [6]

$$\frac{df}{dt} = \alpha(\rho - f) \tag{1}$$

where α is the rate at which agents reevaluate their resource choice and ρ is the probability that an agent will prefer resource 1 over 2 when it makes a choice. Generally, ρ is a function of f through the density dependent payoffs. In terms of the payoffs and uncertainty, we have

$$\rho = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{G_1(f) - G_2(f)}{2\sigma}\right)\right) \tag{2}$$

where σ quantifies the uncertainty. Notice that this definition captures the simple requirement that an agent is more likely to prefer a resource when its payoff is relatively large. Finally, delays in information are modeled by supposing that the payoffs that enter into ρ at time t are the values they had at a delayed time $t - \tau$.

For a typical system of many agents with a mixture of cooperative and competitive payoffs, the kinds of dynamical behaviors exhibited by the model are shown in Fig. 1. When the delays and uncertainty are fairly small, the system converges to an equilibrium point close to the optimal obtainable by an omniscient, central controller. As the information available to the agents becomes more corrupted, the equilibrium point moves further from the optimal value. With increasing delays, the equilibrium eventually becomes unstable, leading to the oscillatory and chaotic behavior shown in the figure. In these cases, the number of agents using particular resources continues to vary so that the system spends relatively little time near the optimal value, with the consequent drop in its overall performance.

3 Rewarding Performance

We now describe an effective procedure for controlling chaos in distributed systems. It is based on a mechanism that rewards agents according to their actual performance. As we shall see, such an algorithm leads to the emergence of a diverse community of agents out of an essentially homogenous one. This diversity in turn eliminates chaotic behavior through a series of dynamical bifurcations which render chaos a transient phenomenon.

The actual performance of computational processes can be rewarded in a number of ways. A particularly appealing one is to mimic the mechanism found in biological evolution, where fitness determines the number of survivors of a given species in a changing environment. This mechanism is used in computation under the

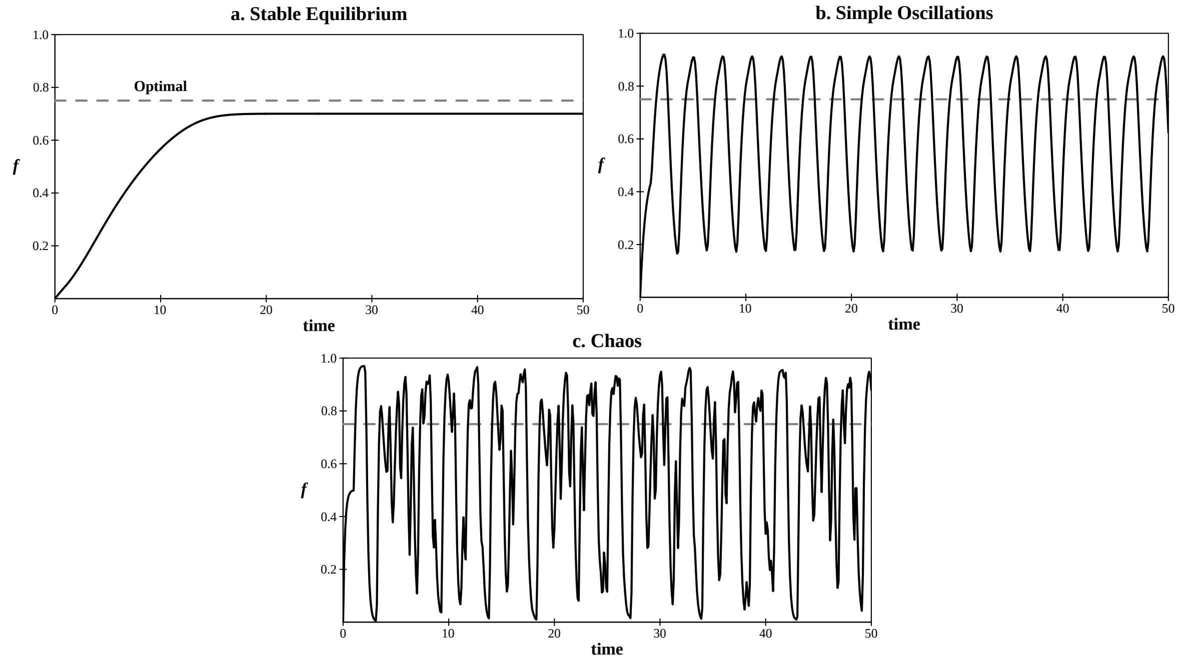

Figure 1: Typical behaviors for the fraction f of agents using resource 1 as a function of time for successively longer delays: a) relaxation toward stable equilibrium, b) simple persistent oscillations, and c) chaotic oscillations. The payoffs are $G_1 = 4 + 7f - 5.333f^2$ for resource 1 and $G_2 = 4 + 3f$ for resource 2. The time scale is in units of the delay time τ, $\sigma = 1/4$ and the dashed line shows the optimal allocation for these payoffs.

name of *genetic algorithms* [4], though often in the case in which agents interact with a fixed environment rather than each other. Another example is provided by computational systems modelled on ideal economic markets [9, 11], which reward good performance in terms of profits. In this case, agents pay for the use of resources, and they in turn are paid for completing their tasks. Those making the best choices collect the most currency and are able to outbid others for the use of resources. Consequently they come to dominate the system.

While there is a range of possible reward mechanisms, their net effect is to increase the proportion of agents that are performing successfully, thereby decreasing the number of those who do not do as well. It is with this insight in mind that we now develop a general theory of effective reward mechanisms without resorting to the details of their implementations. Since this change in agent mix will in turn change the choices made by every agent and their payoffs, those that were initially most successful need not be so in the future. This leads to an evolving diversity whose eventual stability is by no means obvious.

Before proceeding with the theory we point out that the resource payoffs that we will consider are instantaneous ones (i.e., shorter than the delays in the system), e.g., work actually done by a machine, currency actually received, etc. Other reward mechanisms, such as those based on averaged past performance, could lead to very different behavior from the one exhibited in this paper.

In order to investigate the effects of rewarding actual performance we generalize the previous model of computational ecosystems by allowing agents to be of different types, which gives them different performance characteristics. Recall that the agents need to estimate the current state of the system based on imperfect and delayed information in order to make good choices. This can be done in a number of ways, ranging from extremely simple extrapolations from previous data to complex forecasting techniques. The different types of agents then correspond to the various ways in which they can make these extrapolations.

Within this context, a computational ecosystem can be described by specifying the fraction of agents, f_{rs} of a given type s using a given resource r at a particular time. We will also define the total fraction of agents using a resource of a particular type as

$$f_r^{\text{res}} = \sum_s f_{rs} \tag{3}$$

$$f_s^{\text{type}} = \sum_r f_{rs}$$

respectively.

As mentioned previously, the net effect of rewarding performance is to increase the fraction of highly performing agents. If γ is the rate at which performance is rewarded, then Eq. (1) is enhanced with an extra term that corresponds to this reward mechanism. This gives

$$\frac{df_{rs}}{dt} = \alpha \left(f_s^{\text{type}} \rho_{rs} - f_{rs} \right) + \gamma \left(f_r^{\text{res}} \eta_s - f_{rs} \right) \tag{4}$$

where the first term is analogous to that of the previous theory, and the second term incorporates the effect of rewards on the population. In this equation ρ_{rs} is the probability that an agent of type s will prefer resource r when it makes a choice, and η_s is the probability that new agents will be of type s, which we take to be proportional to the actual payoff associated with agents of type s. As before, α denotes the rate at which agents make resource choices and the detailed interpretation of γ depends on the particular reward mechanism involved. For example, if they are replaced on the basis of their fitness it is the rate at which this happens. In a market system, on the other hand, γ corresponds to the rate at which agents are paid. Notice that in this case, the fraction of each type is proportional to the wealth of agents of that type.

Since the total fraction of agents of all types must be one, a simple form of the normalization condition can be obtained if one considers the relative payoff, which is given by[1]

$$\eta_s = \frac{\sum_r f_{rs} G_r}{\sum_r f_r^{\text{res}} G_r} \tag{6}$$

Note that the numerator is the actual payoff received by agents of type s given their current resource usage and the denominator is the total payoff for all agents in the system, both normalized to the total number of agents in the system.

Summing Eq. (4) over all resources and types gives

$$\begin{aligned} \frac{df_r^{\text{res}}}{dt} &= \alpha \left(\sum_s f_s^{\text{type}} \rho_{rs} - f_r^{\text{res}} \right) \\ \frac{df_s^{\text{type}}}{dt} &= \gamma \left(\eta_s - f_s^{\text{type}} \right) \end{aligned} \tag{7}$$

which describe the dynamics of overall resource use and the distribution of agent types, respectively. Note that this implies that those agent types which receive greater than average payoff (i.e., types for which $\eta_s > f_s^{\text{type}}$) will increase in the system at the expense of the low performing types.

Note that the actual payoffs can only reward existing types of agents. Thus in order to introduce new variations into the population an additional mechanism is needed (e.g., corresponding to mutation in genetic algorithms or learning).

4 Results

In order to illustrate the effectiveness of rewarding actual payoffs in controlling chaos, we examine the dynamics generated by Eq. (4) for the case in which agents choose among two resources with cooperative payoffs, a case which we have shown to generate chaotic behavior in the absence of rewards [7]. As in the particular example of Fig. 1c, we use $\tau = 10$, $G_1 = 4 + 7f_1 - 5.333f_1^2$, $G_2 = 7 - 3f_2$, $\sigma = 1/4$ and an initial condition in which all agents start by using resource 2.

[1] This form assumes positive payoffs, e.g., they could be growth rates. If the payoffs can be negative (e.g., they are currency changes in an economic system), one can use instead the difference between the actual payoffs and their minimum value m. Since the η_s must sum to 1, this will give

$$\eta_s = \frac{\sum_r f_{rs} G_r - m}{\sum_r f_r^{\text{res}} G_r - Sm} \tag{5}$$

which reduces to the previous case when $m = 0$.

One kind of diversity among agents is motivated by the simple case in which the system oscillates with a fixed period. In this case, those agents that are able to discover the period of the oscillation can then use this knowledge to reliably estimate the current system state in spite of delays in information. Notice that this estimate does not neccesarily guarantee that they will keep performing well in the future, for their choice can change the basic frequency of oscillation of the system.

In what follows, we take the diversity of agent types to correspond to the different past horizons, or extra delays, that they use to extrapolate to the current state of the system. These differences in estimation could be due to having a variety of procedures for analyzing the system's behavior. Specifically, we identify different agent types with the different assumed periods which range over a given interval. Thus, we take agents of type s to use an effective delay of $\tau + s$ while evaluating their choices.

The resulting behavior is shown in Fig. 2 which should be contrasted with Fig. 1c. We used an interval of extra delays ranging from 0 to 40. As shown, the introduction of actual payoffs induces a chaotic transient which, after a series of dynamical bifurcations, settles into a fixed point that signals stable behavior. Furthermore, this fixed point is exactly that obtained in the case of no delays. That this equilibrium is stable against perturbations can seen by the fact that if the system were perturbed again (as shown in Fig. 3), it rapidly returns to its previous value. In additional experiments, with a smaller range of delays, we found that the system continued to oscillate without achieving the fixed point.

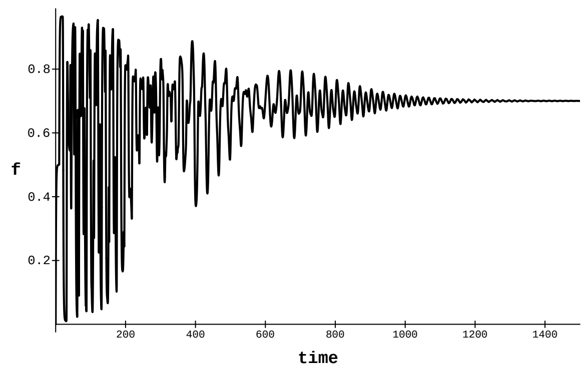

Figure 2: Fraction of agents using resource 1 as a function of time with adjustment based on actual payoff. These parameters correspond to Fig. 1c so without the adjustment the system would remain chaotic.

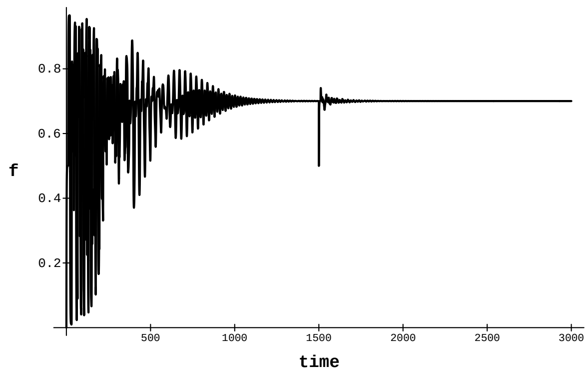

Figure 3: Behavior of the system shown in Fig. 2 with a perturbation introduced at time 1500.

This transient chaos and its eventual stability can be understood from the distribution of agents with extra delays as a function of time. As can be seen in Fig. 4 actual payoffs lead to a highly heterogeneous system, characterized by a diverse population of agents of different types. It also shows that the fraction of agents with certain extra delays increases greatly. These delays correspond to the major periodicities in the system.

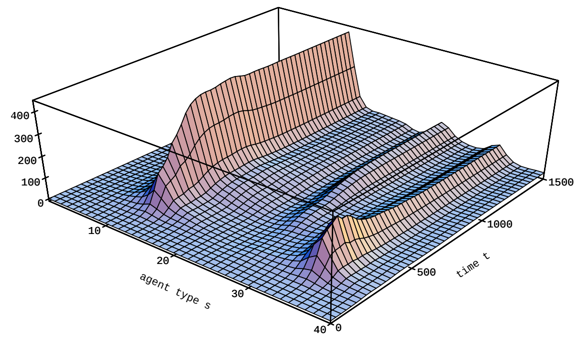

Figure 4: Ratio $f_s^{\rm type}(t)/f_s^{\rm type}(0)$ of the fraction of agents of each type, normalized to their initial values, as a function of time. Note there are several peaks, which correspond to agents with extra delays of 12, 26 and 34 time units. Since $\tau = 10$, these match periods of length 22, 36 and 44 respectively.

5 The Value of Diversity

As we showed in the previous section, rewarding the performance of large collections of agents engaging in resource choices leads to a highly diverse mix of agents that stabilize the system. This suggests that the real cause of stability in a distributed system is sufficient diversity, and that the reward mechanism is just a way of automatically finding a good mix.

This raises two interesting questions: 1) under what conditions does a stable heterogeneous system exist, and 2) what kinds of reward mechanisms generate a stable mix. We now proceed to answer them both.

5.1 The Right Mix

The stability of a system is determined by the behavior of a perturbation around equilibrium, which can be found from the linearized version of Eq. (4). Since heterogeneous systems are known to be more stable than homogeneous ones [7], the important question to resolve is the amount of diversity needed to make the system stable.

In our case, the diversity is related to the range of different delays that agents can have. For a continuous distribution of extra delays, the characteristic equation governing stability is obtained by assuming a solution of the type $e^{\lambda t}$ in the linearized equation, giving

$$\lambda + \alpha - \alpha\rho' \int ds\, f(s) e^{-\lambda(s+\tau)} = 0 \tag{8}$$

Values of λ that satisfy this equation correspond to solutions. Thus, stability requires that all such λ have negative real parts, so that perturbations will relax back to equilibrium. As an example, suppose agent types are uniformly distributed in $(0, S)$. Then $f(s) = 1/S$, and the characteristic equation becomes

$$\lambda + \alpha - \alpha\rho' \frac{1 - e^{-\lambda S}}{\lambda S} e^{-\lambda\tau} = 0 \tag{9}$$

Defining a normalized measure of the diversity of the system for this case by $\eta \equiv S/\tau$, introducing the new variable $z \equiv \lambda\tau(1+\eta)$, and multipling Eq. (9) by $\tau(1+\eta)ze^z$ introduces an extra root at $z = 0$ and gives

$$(z^2 + az)e^z - b + be^{rz} = 0 \tag{10}$$

where

$$\begin{aligned} a &= \alpha\tau(1+\eta) > 0 \\ b &= -\rho' \frac{\alpha\tau(1+\eta)^2}{\eta} > 0 \\ r &= \frac{\eta}{1+\eta} \in (0,1) \end{aligned} \tag{11}$$

The stability of the system with uniform distribution of agents with extra delays thus reduces to finding the condition under which all roots of Eq. (10), other than $z = 0$, have negative real parts. This equation is a particular instance of an *exponential polynomial*, whose terms consist of powers multiplied by exponentials. Unlike regular polynomials, these objects generally have an infinite number of roots, and are important in the study of the stability properties of differential-delay equations [1]. Established methods can then be used to determine when they have roots with positive real parts. This in turn defines the stability boundary of the equation. The result for the particular case in which $\rho' = -3.41044$, corresponding to the parameters used in §4, is shown in of Fig. 5a.

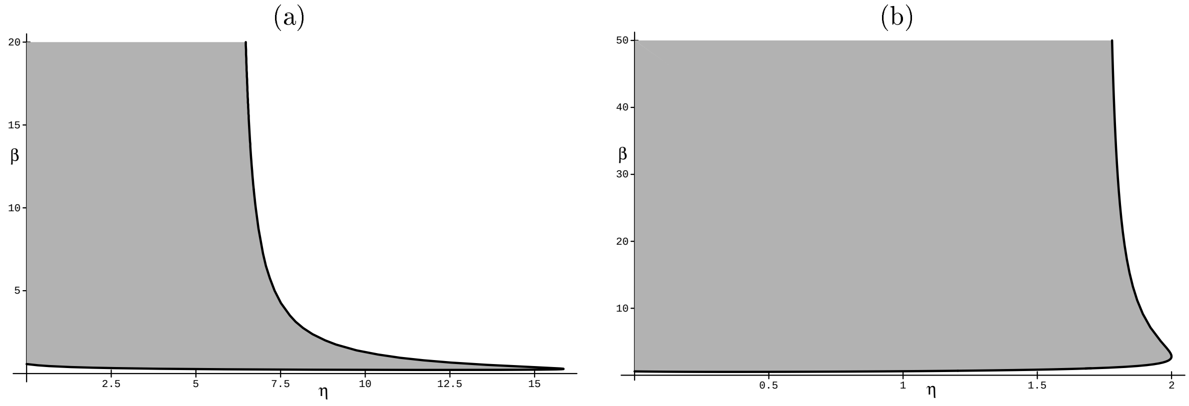

Figure 5: Stability as a function of $\beta = \alpha\tau$ and $\eta = S/\tau$ for two possible distributions of agent types: a) $f(s) = 1/S$ in $(0, S)$, and b) $f(s) = (1/S)e^{-s/S}$. The system is unstable in the shaded regions and stable to the right and below the curves.

Similarly, if we choose an exponential distribution of delays, i.e., $f(s) = (1/S)e^{-s/S}$ with positive S, the characteristic equation acquires the form

$$(z^2 + pz + q)e^z + r = 0 \tag{12}$$

where

$$\begin{aligned} p &= \alpha\tau + \frac{1}{\eta} > 0 \\ q &= \frac{\alpha\tau}{\eta} > 0 \\ r &= -\frac{\alpha\tau\rho'}{\eta} > 0 \end{aligned} \tag{13}$$

and $z \equiv \lambda\tau$. An analysis similar to that for the uniform distribution case leads to the stability diagram shown in Fig. 5b.

Although the actual distributions of agent types can differ from these two cases, the similarity between the stability diagrams suggests that regardless of the magnitude of β one can always find an appropriate mix that will make the system stable. This property follows from the vertical asymptote of the stability boundary. It also illustrates the need for a minimum diversity in the system in order to make it stable when the delays aren't too small.

Having established the right mix that produces stability one may wonder whether a static assignment of agent types at an initial time would not constitute a simpler and more direct procedure to stabilize the system without resorting to a dynamic reward mechanism. While this is indeed the case in a non-fluctuating environment, such a static mechanism cannot cope with changes in both the nature of the system (e.g., a change in problem requirements that affects the payoff of various choices) and the arrival of new tasks. It is precisely to avoid this vulnerability by keeping the system adaptive that a dynamic procedure is needed.

5.2 The Right Reward

Having seen how sufficient diversity stabilizes a distributed system, we now turn to the mechanisms that can generate such heterogeneity, as well as the time that it takes for the system to stabilize. In particular, the details of the reward procedures determine whether the system can even find a stable mix of agents. In the cases describe above, reward was proportional to actual performance, as measured by the payoffs associated with the resources used. One might also wonder whether stability would be achieved more rapidly by giving greater (than their fair share) increases to the top performers.

We have examined two such cases: a) rewards proportional to the square of their actual performance, and b) giving all the rewards to top performers (e.g., those performing at the 90th percentile or better in the population). In the former case we observed stability with a shorter transient, whereas in the latter case the mix of agents continued to change through time, thus preventing stable behavior. This can be understood in terms of our earlier observation that whereas a small percentage agents can identify oscillation periods and thereby reduce their amplitude, a large number of them can no longer perform well.

Note that the time to reach equilibrium is determined by two parameters of the system. The first is the time that it takes to find a stable mix of agent types, which is governed by γ, and the second the rate at which perturbations relax, given the stable mix. The latter is determined by the largest real part of any of the roots, λ, of the characteristic equation.

5.3 Generating Diversity

Up to this point, we have considered the specific situation of a collection of agents with different delays in their appraisal of the system evolution. As we showed, the development of a heterogenous population leads to stability. It is of interest to inquire whether using rewards to increase diversity works more generally than in the case of extra delays.

That this is the case in a more general sense is suggested by the following. Suppose there is a variety of techniques for evaluating the state of the system, and that each of these techniques is used by a particular type of agent. In the previous case, these techniques differed in the amount of extra delays used in the evaluation of the system state. Consider now the case where these techniques involve the use of the same delay but make different systematic errors or biases in the evaluations. For instance, a positive bias would correspond to a heuristic that tends to overvalue a resource. In this case, the probability that an agent of type s will choose resource 1 is given by,

$$\rho_{1s} = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{G_1(f_1(t-\tau)) + s - G_2(f_1(t-\tau))}{2\sigma}\right)\right) \tag{14}$$

where s is the net bias by which resource 1 is preferred over resource 2.

The dynamics generated by such a system is shown in Fig. 6. For these parameters, the system is chaotic in the absense of rewards, as shown in Fig. 6a. The introduction of rewards however, rapidly stabilizes the system through a set of dynamical bifurcations that lead to a fixed point. Moreover, this fixed point is optimal in the sense that it corresponds to the resource usage obtained when there are no delays or uncertainty in the system. The diversity generated by this dynamical reward as a function of time is shown in Fig. 7. Notice how the extreme bias values gained the most.

As this example shows, the ability to control chaos in distributed systems through a reward mechanism extends to inhomogeneities beyond those characterized by different delays. These can also include different preferences on the part of the agents for particular resources, or preferences for computing rapidly and expensively, versus slowly and cheaply.

6 Conclusion

In this paper we showed how a simple mechanism based on the actual performance of agents in a distributed computational system can stabilize an otherwise chaotic system. This leads in turn to greatly improved system performance, which in some cases approaches the optimal one that would be obtained by an omniscient central controller.

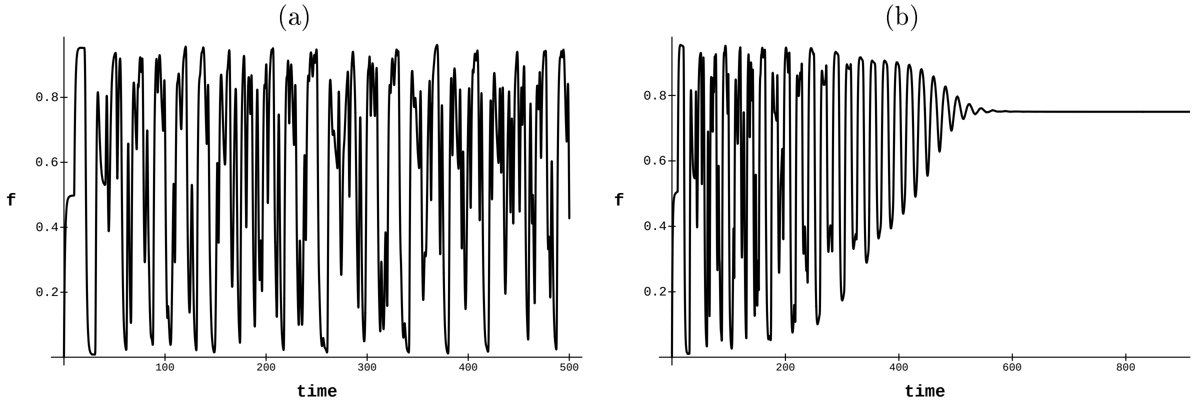

Figure 6: Fraction of agents using resource 1 for a collection of biased agents with (a) no reward and (b) with reward. Here the system stabilizes at a value of 0.750047, which differs from the equilibrium point (for this value of uncertainty) in the absence of delay of 0.700318. Thus, it reaches the optimal value corresponding to zero uncertainity (i.e., the point where $G_1(f) = G_2(f)$). Here $\beta = 10$ and $\gamma = 1$.

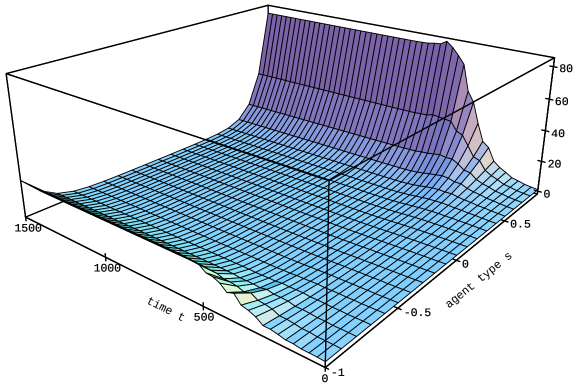

Figure 7: Ratio $f_s^{\text{type}}(t)/f_s^{\text{type}}(0)$ of the fraction of agents of different types as a function of time normalized to their initial values.

This stability is achieved by making chaos a transient phenomena. In essence, the addition of this reward mechanism has the effect of dynamically changing the control parameters of the system dynamics in such a way that a global fixed point of the system is achieved. This raises the issue of the length of the chaotic transient as compared to the time needed for most agents to complete their tasks. Even when the transients are long, the results of this study show that the range of variation gradually decreases, thereby improving performance even before the fixed point is achieved.

A particularly relevant question is the extent to which these results generalize beyond the assumptions of this paper. Since we considered agents selecting between only two choices or resources, it is important to understand what happens when there are many resources the agents can choose from. One may argue that since diversity is the key to stability, a plurality of resources provides enough channels to develop the necessary heterogeneity, which is what we observed in situations with three resources. Another note of caution has to do with the effect of fluctuations on a finite population of agent types. While we have shown that sufficient diversity can, on average, stabilize the system, in practice a fluctuation could wipe out those agent types that would otherwise be successful in stabilizing the system. Thus we need either a large number of each kind of agent or a mechanism, such as mutation, to create new kinds of agents.

One problem that one may encounter when trying to apply these results to general computational and market systems has to do with individual performance measures. In any heterogeneous system, agents can have very different kinds of preferences (megaflops, currency, prestige, etc.) which are difficult to reduce to a single scalar. In our theory, we avoided this problem by assuming all payoffs are comparable so that we could reward the agents based on a simple comparison of their actual payoffs. This amounts to a single measure of fitness for the agents' performance. Although the applicability of this method to more general situations remains an open problem, we note that economic systems provide a universal currency system that reduces everything to a single quantity, to which our reward mechanism can then be applied.

A final issue concerns the time scales over which rewards are assigned to agents. In our treatment, we assumed the rewards were always based on the performance at the time they were given. Since in many cases this procedure is delayed, there is the question of the extent to which rewards based on past performance are also able to stabilize chaotic distributed systems.

In spite these issues, we believe that the control mechanism we introduced not only stabilizes a distributed system, but also provides a dynamical formalism for analyzing the behavior of complex markets.

References

[1] R. Bellman and K. L. Cooke. *Differential-Difference Equations.* Academic Press, New York, 1963.

[2] Harry R. Erwin. Mixing and sensitive dependence on initial conditions in computer systems. *CMG Transactions*, 65:3–6, 1989.

[3] Les Gasser and Michael N. Huhns, editors. *Distributed Artificial Intelligence*, volume 2. Morgan Kaufmann, Menlo Park, CA, 1989.

[4] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, NY, 1989.

[5] B. A. Huberman and E. Lumer. Dynamics of adaptive systems. *IEEE Trans. on Circuits and Systems*, CAS-37:547–550, April 1990.

[6] Bernardo A. Huberman and Tad Hogg. The behavior of computational ecologies. In B. A. Huberman, editor, *The Ecology of Computation*, pages 77–115. North-Holland, Amsterdam, 1988.

[7] J. O. Kephart, T. Hogg, and B. A. Huberman. Dynamics of computational ecosystems. *Physical Review A*, 40:404–421, 1989.

[8] J. O. Kephart, T. Hogg, and B. A. Huberman. Collective behavior of predictive agents. *Physica D*, 42:48–65, 1990.

[9] Mark S. Miller and K. Eric Drexler. Markets and computation: Agoric open systems. In B. A. Huberman, editor, *The Ecology of Computation*, pages 133–176. North-Holland, Amsterdam, 1988.

[10] S. Sinha, R. Ramaswamy, and J. S. Rao. Adaptive control in nonlinear dynamics. *Physica D*, 43:118–128, May 1990.

[11] Carl A. Waldspurger, Tad Hogg, Bernardo A. Huberman, Jeffery O. Kephart, and W. Scott Stornetta. Spawn: A distributed computational economy. *IEEE Trans. on Software Engineering*, 18(2):103–117, February 1992.