HP Labs' Complex Adaptive Systems Group Research Overview†

Andrew Byde, Dave Cliff, Matthew Williamson
Digital Media Systems Laboratory
HP Laboratories Bristol
HPL-2004-79
May 5, 2004*

E-mail: firstname_lastname@hp.com

virus(es), throttle, throttling, markets, market-based control, markets, trader(s), trading, agent(s), evolution, complex adaptive system(s), self-organization, self-managing

The Complex Adaptive Systems (CAS) group at HP Labs, Bristol, was formed in November 2001 to study the science and engineering of complex, dynamic, parallel, distributed, adaptive systems, so far as they are relevant to HP's present and future business strategy. Such systems can exhibit highly desirable characteristics of resilient self-organisation, self-regulation, self-healing, and adaptation over multiple spatial and temporal scales, and can also be used for complex optimisation and automated-design tasks. CAS aims to grow world-class expertise for HP in the theory and practice of artificial systems that exhibit these desirable characteristics. This paper is an overview of CAS research projects relevant to self-configuring, self-organizing, self-managing and self-repairing IT systems.

* Internal Accession Date Only
†SELF-STAR International Workshop on Self-*properties of Complex Information Systems, Bologna, Italy, 31 May – 2 June 2004
Approved for External Publication

HP Labs' Complex Adaptive Systems Group Research Overview

Andrew Byde, Dave Cliff & Matthew Williamson

Prepared for the SELF-STAR International Workshop on Self-* properties of Complex Information Systems, Bologna, Italy, 31 May – 2 June 2004.

The Complex Adaptive Systems (CAS) group at HP Labs, Bristol [1], was formed in November 2001 to study the science and engineering of complex, dynamic, parallel, distributed, adaptive systems, so far as they are relevant to HP's present and future business strategy. Such systems can exhibit highly desirable characteristics of resilient self-organisation, self-regulation, self-healing, and adaptation over multiple spatial and temporal scales, and can also be used for complex optimisation and automated-design tasks. CAS aims to grow world-class expertise for HP in the theory and practice of artificial systems that exhibit these desirable characteristics.

Of our current projects, those that are particularly relevant to the question of self-maintaining systems are the project investigating market-based resource allocation for Utility Data Centres, and the projects exploring throttling of malicious code (e.g. viruses) and unwanted data (e.g. spam).

Virus Throttling

We are investigating benign and biologically-inspired methods for ameliorating the effects of "malware" such as computer viruses and worms. The basic problem is well known: telling the difference between malicious and benign messages/behaviour is very difficult for computers. The current typical systemic mode of response to infection is for the computer to prevent the spread of anything whose signature is known, and to wait for humans to recognize and describe the signatures of any new threats that emerge. The evident problem with this approach is that the human response loop in question works on timescales of days or weeks that take far too long to be effective. By the time a system manager has discovered why her computers have gone down and an effective patch or signature to distribute to others has been designed, the virus has typically done its work already, spreading itself to thousands or millions of other computers. What is needed is a way for computers to "look after themselves".

The "throttling" solution to this problem is simple but effective [2]. The approach relies on the observation that the normal patterns of network traffic (messages, packets) on many protocols are quite different from the traffic generated by a spreading virus, with the virus contacting many different machines at a high rate. To limit propagation a rate-limiter or virus throttle is enabled that does not affect normal traffic, but quickly slows and stops viral traffic. The approach prevents an infected machine spreading the virus further, although it does not prevent the machine from being infected in the first place. Thus the method limits attacks at the *system* level, not at the individual machine level, by restricting computers so that they can only spread the infection at an extremely low rate. This directly addresses the two ways that viruses cause damage: fewer machines spreading the virus will reduce the number of machines infected and reduce the traffic generated by the virus.

The throttle limits the rate that a machine can interact with different machines. A machine is determined to be "different" if its address is not contained in a short history list maintained by the throttle. If the address is in the list, the message is passed without delay; but if it is not, the message is queued. The queue is serviced regularly (e.g. once per second), removing messages, sending them and updating the history. The queue mechanism thus ensures that for example, the machine can interact with at most one new machine per second.

Since most normal traffic is at a low rate and to a slowly varying set of machines, this rate limiter (when set with suitable values for parameters such as the rate limit and the size of the history list) will not affect normal traffic greatly. However, if a virus were to attempt to spread faster than allowed, it would be forced to spread only at the allowed maximum rate. In practice, worms and viruses attempt to connect to hundreds of machines per second, so the pending queue gets large very quickly. This can be easily detected and the further propagation of the virus stopped completely.

These techniques have been implemented for both IP traffic [3], [2] and for email [4]. Work is currently underway applying the idea to instant messaging [5].

Spam Control

Spam or junk mail is a growing problem for today's enterprises. MessageLabs (an Internet email handler) measures that 55% of all mail is spam and predicts that this will grow to around 70% by the end of 2004 [6], [7]. The most obvious effect of this junk mail is to clutter users' inboxes, but is has another effect: the mail infrastructure cannot keep up with the increase in mail, and is becoming overloaded. This has the effect of poor service (e.g. transit delays) for email. Part of the reason for this is the extra processing that is now required to detect viruses and spam in email. The "scanner" is the bottleneck in the mail server and if an underpowered machine is used, delays will result. As mail volumes increase, machines rapidly become underpowered!

We [8] have analysed large volumes of email traffic and deduced that it is possible to accurately predict before a mail is completely received whether or not it is likely to be junk (spam, virus or undeliverable) or not. This prediction can then be used to prioritise good mail through the mail server, so that transit delays are reduced and the quality of service improved. The effects can be large: for mail that would be delayed by over 4 hours, the prioritisation scheme gives average delays of only 22 seconds.

The prediction is based on sending history – a mail server on the Internet tends to send the same sort of mail i.e. one sending junk mail will likely continue and vice versa. As messages are received and scanned the accuracy of the prediction improves, although only a small number of messages (less than 10) are required for the prediction to converge.

This scheme gives resilience back to the mail server, allowing it to cope with large mail loads while maintaining good service. By maintaining the prediction, the mail server optimises its own resource allocation depending on the traffic flowing through it. In addition, the mail server can be provisioned to carry the volume of good mail, and any increases in the volume of spam mail will not impact the operation of the server.

We are currently analysing how this mechanism could be used to reduce the overall volume of spam processed, as well as ensure that good mail is processed promptly.

Market-based Control of Utility Data Centres

Another focus of the HP Labs CAS group that is of particular relevance to the issue of self-managed systems is our ongoing investigations into the possibility that a managed IT service provider who seeks to run multiple demand-varying services on a common infrastructure such as HP's Utility Data Centre (UDC) product, might choose to organize the allocation of resources between competing services as a market – so-called market-based control (MBC).

In this framework, once service contracts have been signed between the service customer (e.g. an enterprise wishing to outsource payroll management), and the computing fabric owner (who will run the relevant applications), the customer need not concern themselves with resource requirements, and the fabric owner need only attempt to maximize the return on their Service Level Agreement (SLA) by varying the allocation of resources given to each service. This can be done in many ways; our proposal is to assign an autonomous management agent to each service, and

to give the agent internal currency in proportion to the performance of its service with respect to the relevant SLA. This currency can them be redeemed in an internal market for the various compute resources that the fabric owner has, such as storage, CPU cycles, bandwidth, etc. In this way, the fabric provider only makes budgeting decisions for the various service management agents, and lets the market take care of assigning resources where they are needed most, as measured by the agents' willingness to pay for them.
We have studied experimental simulations of markets of this sort for regulating resource allocation between idealized job-processing workflows, and have found that the pricing mechanism can indeed be an effective method of self-management [9]. Agents predict the effect on their workflow's queues of having various resource levels, and the consequent impact that will have on their income, and thus their willingness to pay for those resource levels. In general, when correctly set up, the system can manage distribution of resources between jobs as well as any other policy. In some circumstances, for example, we find that prices oscillate in such a way as to implement time-sharing of scarce indivisible resources. However, it must be said that the systemic behaviour is (not surprisingly) highly dependent on the value prediction algorithms at the management agents' disposal: for example, in [9] we describe how an algorithm for predicting job processing times for various resource levels that is *less* precise leads to *greater* efficiency for the system as a whole. The hypothesised reason – that the less precise algorithm has a beneficial damping effect on system volatility – indicates that autonomous multi-agent systems such as the one described probably need to be designed and tested at the system level, not as groups of individual parts.

But the system-level design and testing of autonomous multi-agent systems typically requires skilled human practitioners, and in many instances is more of an art than a science, with trial-and-error methods used given the absence of any established rigorous engineering techniques.

For this reason, the HP Labs CAS group has made significant research investments in developing automated methods for the design and optimisation of autonomous multi-agent systems, focusing particularly on distributed market-based resource allocation and load balancing in large-scale distributed computer systems such as UDCs. Specifically, we have explored the use of evolutionary computation techniques such as genetic algorithms (GAs) to automatically optimise trader-agents and market-mechanisms that could be used in MBC systems. To date, much of our work has concentrated on GA-optimisation of markets populated by software agents running the "ZIP" trader-agent algorithm [10] which was developed initially at HP Labs and subsequently demonstrated by researchers at IBM to outperform human traders [11].

Two desirable system-level behaviours of the agent-based markets in an MBC system are that the agents' transaction prices rapidly converge on the underlying equilibrium price and that the convergence is stable [10]. The price dynamics of markets populated by ZIP traders are determined by the values of eight real-valued control parameters, so any one ZIP-trader market can be characterised as a point in an 8-dimensional real space. In early work [12], we successfully demonstrated that a simple GA could find values for the elements of these 8-dimensional ZIP-market vectors that were better than the manually-set values originally chosen by the designer of the ZIP algorithm, and analysis of the evolutionary dynamics of the GA system demonstrated that this was a non-trivial search problem.

Following IBM's demonstration of the superiority of ZIP traders (and of another trader-agent algorithm known as "MGD") over human traders [11], the IBM research team claimed that it seemed likely that, in future, economically significant online auction-markets (such as those operated by international equity and derivative exchanges) might be depopulated, with the current human traders being replaced by automated software-agent traders. This prompted us to explore the possibility that, in markets where it is known *a priori* that all the traders are software agents and no humans are present, new forms of market mechanism (i.e. the rules that govern the behaviours of and interactions between the trader-agents) might be discovered that are better (in some sense) than the traditional mechanisms. The "traditional" mechanisms are usually online re-implementations of traditional market mechanisms originally designed by humans and for humans.

For the most economically significant form of auction mechanism (the so-called Continuous Double Auction, or CDA, used in almost all of the world's international financial exchanges), economists remain unable to explain precisely what aspects of the auction mechanism contribute to the observable dynamics of markets organised according to that mechanism. This gives some indication that designing new mechanisms is a non-trivial task, even for a skilled economist, and can again often involve trial-and-error design methods where there is a lack of applicable theoretical or analytic results on which to base a design. As an alternative to manual design, we have explored the automated design of new agent-based market mechanisms again using a GA, this time to explore a continuous space of possible auction mechanisms which includes the CDA mechanism but also includes peculiar hybrid mechanisms readily implementable as online exchanges or marketplaces but unlike any traditional market mechanism. To our surprise, when attempting to find mechanisms which gave the most rapid and most stable convergence of transaction prices to the underlying equilibrium price, it was particular instances of these hybrid markets (and not the CDA) that the GA identified as being best. Although originally motivated by attempting to design better internal agent-based markets for MBC systems applicable to UDCs, this work has now attracted significant attention from the world of international equity traders and exchanges. Although the original work concentrated on ZIP-trader markets, one of our subsequent studies [13] established that the GA could find non-traditional hybrid market mechanisms that were better than the comparable traditional market mechanisms, regardless of the nature of the trader-agents in those markets (i.e. whether the traders are human, or any form of artificial agent, the GA-designed markets are better).

We are now actively exploring the application of these results to the automatic design and optimisation of new trader-agents and market-mechanisms for MBC of UDCs and similar distributed large-scale computing systems.

Conclusion

This paper has given an overview of selected research by the HP Labs CAS group that is relevant to self-star computing. Further details are available on our website [1].

References

1. http://www.hpl.hp.com/research/bicas/.
2. Matthew M Williamson. "Throttling viruses: Restricting propagation to defeat malicious mobile code". In *Proceedings of ACSAC Security Confer-ence*, pages 61 – 68, Las Vegas, Nevada, December 2002. HP Labs Technical report HPL-2002-172.
3. Jamie Twycross and Matthew M. Williamson. "Implementing and testing a virus throttle". In *Proceedings 12th USENIX Security Symposium, 2003*, pages 285 – 294, Washington DC, August 2003. USENIX. HP Labs Technical report HPL-2003-103.
4. Matthew M Williamson. "Design, implementation and test of an email virus throttle". In *Proceedings of ACSAC Security Conference*, Las Vegas, Nevada, December 2003. HP Labs Technical report HPL-2003-118.
5. Matthew M Williamson, Alan Parry and Andrew Byde. "Virus Throttling for Instant Messaging." To appear at *Virus Bulletin Conference 2004*, Chicago IL.
6. MessageLabs. MessageLabs Monthly View, November 2003. Published on the MessageLabs site, http://www.messagelabs.com.
7. MessageLabs, "Spam and viruses hit all time highs in 2003", December 2003. Published on the MessageLabs site, http://www.messagelabs.com .
8. D. Twining, M. Williamson, M. Mowbray, & M. Rohmouni, "Email Prioritization: reducing delays on legitimate mail caused by junk mail". To appear at *USENIX Annual Technical Conference*, June 2004, Boston MA. HP Labs Technical Report HPL-2004--5.
9. A. Byde, M. Salle, & C Bartolini, "Market-Based Resource Allocation for Utility Data Centers". HP Labs Technical Report HPL-2003-188.
10. D. Cliff, "Minimal-intelligence agents for bargaining behaviours in market environments". HP Labs Technical Report HPL-97-91.
11. R. Das, J. Hanson, J. Kephart, & G. Tesauro, "Agent-human interactions in the continuous double auction". In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-01)*, 2001.

12. D. Cliff, “Evolutionary optimization of parameter sets for adaptive software-agent traders in continuous double-auction markets”. Presented at *The Artificial Societies and Computational Markets (ASCMA98) workshop at the Second International Conference on Autonomous Agents*, May 1998. HP Labs Technical Report HPL-2001-99.
13. A. Byde, “Applying Evolutionary Game Theory to Auction Mechanism Design”. Presented at the *ACM Conference on E-Commerce*, 2003. HP Labs Technical Report HPL-2002-321.
14. D. Cliff, "Explorations in evolutionary design of online auction market mechanisms". In *Journal of Electronic Commerce Research and Applications.* 2(2):162-175, 2003.