Towards evolution-based autonomy in large-scale systems

Damien Anderson
University of Strathclyde
Glasgow, United Kingdom
damien.anderson@strath.ac.uk

Paul Harvey
Rakuten Mobile
Tokyo, Japan
paul.harvey@rakuten.com

Yusaku Kaneta*
Rakuten Mobile
Tokyo, Japan
yusaku.kaneta@rakuten.com

Petros Papadopoulos
University of Strathclyde
Glasgow, United Kingdom
petros.papadopoulos@strath.ac.uk

Philip Rodgers
Rakuten Mobile
Glasgow, United Kingdom
philip.rodgers@rakuten.com

Marc Roper
University of Strathclyde
Glasgow, United Kingdom
marc.roper@strath.ac.uk

ABSTRACT

To achieve truly autonomous behaviour systems need to adapt to not only previously unseen circumstances but also previously *unimagined* ones. A hierarchical evolutionary system is proposed which is capable of displaying the emergent behaviour necessary to achieve this goal. This paper reports on the practical challenges encountered in implementing this proposed approach in a large-scale open-source system.

1 INTRODUCTION

In "Evolutionary Autonomous Networks" [3], Harvey et al. present their vision for an autonomous network utilising online evolution to achieve guided emergent behaviour. Their approach presents a radical strategy to improve a network's ability to cope with uncertainty and change. The aim of this paper is to present the initial steps we have taken towards realising this vision by examining how evolutionary-based autonomy may be introduced into an example large-scale system–a content delivery network (CDN). This goal is not without its challenges as there are significant hurdles imposed by the design and implementation of the CDN which must be overcome.

2 AN EVOLUTIONARY CDN

The design of the idealised evolutionary autonomous network is illustrated in figure 1. At the topmost level is the Master Evolution Controller which is dedicated to achieving and maintaining a high level of utility (fitness) similar to a service-level agreement. This is achieved through composing a number of Meta-Evolutionary Controllers which focus on less abstract utilities such as network latency, response etc., and these in turn achieve their goals by composing a set of Local Evolution Controllers (LEC) which are dedicated to more concrete aspects of the system such as cache performance or traffic shaping. At the lowest level are Operation Controllers (OC) - these directly control network elements such as load balancing or job scheduling. The key feature of this hierarchy is that it is only the lowest level that is domain specific – the three layers of evolutionary controller above this are all generic modules which may theoretically be applied to any system.

To investigate the practical feasibility of realising this vision, and the ensure it is generic enough to be applicable to other systems,

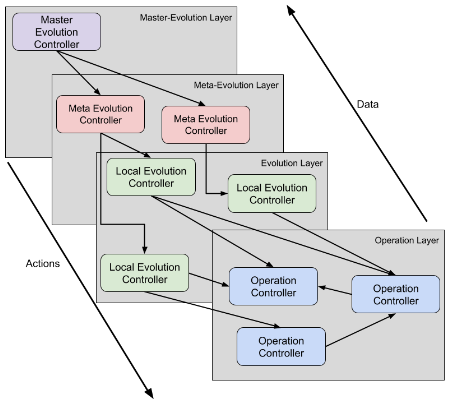
Master-Evolution Layer
Master Evolution Controller
Meta-Evolution Layer
Meta Evolution Controller
Meta Evolution Controller
Evolution Layer
Local Evolution Controller
Local Evolution Controller
Local Evolution Controller
Operation Layer
Operation Controller
Operation Controller
Operation Controller
Data
Actions

Figure 1: Controller Hierarchy (adapted from [3])

we are exploring how evolution can be introduced into a CDN (we chose the open-source system *Varnish*[1] to experiment with) in order to adapt to unforeseen (and unimagined) scenarios. The choice of system type was motivated by the fact that CDNs are an important application for 5G and Beyond 5G networks, and network operators and users would benefit from autonomous cooperation between a CDN and an autonomous network.

A CDN is a network of geographically distributed servers which work together to provide efficient and fast delivery of internet content [1]. In its basic form, a CDN is made up of two components; origin servers (where the content or services are stored) and caching servers (which are responsible for mirroring some or all of the content on the origin servers). Distributing content across the network relative to the end users ensures that content can be delivered more efficiently and quickly (as it can be retrieved from a local cache rather than the potentially more distant and slower origin server) and a move even distribution of network load.

A CDN may be configured in multiple different ways, depending on the nature of the content, requirements of the service, etc. One of the objectives of this project is to explore how evolutionary strategies can autonomously configure and adapt the CDN without knowledge of the domain of operation.

*This work was conducted as a part of Beyond5G R&D Promotion Project (#01701), supported by National Institute of Information and Communication Technology (NICT), Japan.

[1]https://www.varnish-software.com/solutions/cdn/

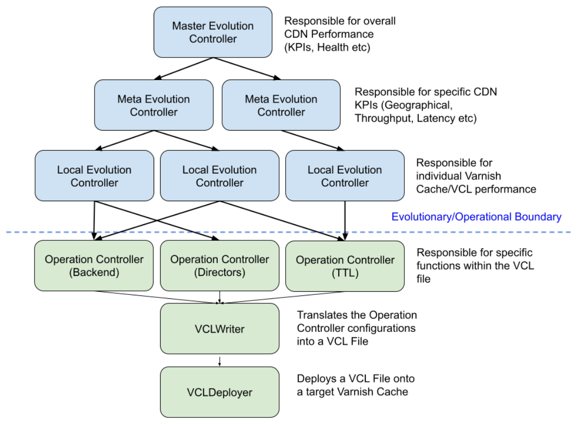
Master Evolution Controller
Responsible for overall CDN Performance (KPIs, Health etc)
Meta Evolution Controller
Meta Evolution Controller
Responsible for specific CDN KPIs (Geographical, Throughput, Latency etc)
Local Evolution Controller
Local Evolution Controller
Local Evolution Controller
Responsible for individual Varnish Cache/VCL performance
Evolutionary/Operational Boundary
Operation Controller (Backend)
Operation Controller (Directors)
Operation Controller (TTL)
Responsible for specific functions within the VCL file
VCLWriter
Translates the Operation Controller configurations into a VCL File
VCLDeployer
Deploys a VCL File onto a target Varnish Cache

Figure 2: Proposed Autonomous System for Varnish CDN

3 IMPLEMENTATION CHALLENGES

3.1 Evolution and Content Delivery Networks

The proposed evolutionary system was designed with autonomous networks in mind, where the periphery of the system, and the responsibility of the operation controllers, is composed of distinct entities such as antennae, often running autonomous collect-analyse-decide-act (CADA) control loops to keep the system operating at optimal conditions. The components of a CDN are not so cleanly separable and so the first major challenge was to identify the domain boundary and the constituents of the LECs and OCs.

By making the LECs responsible for an individual cache within the CDN, the OCs then become elements which may be configured to influence (and optimise) the performance of this cache – backend (origin) servers, directors (scheduling strategies), etc.. LECs then monitor the utility of an existing cache configuration and are also tasked with generation of new solutions (via mutation) that are passed on to the lower level for evaluation.

The mapping of the idealised design described above to the Varnish CDN is illustrated in figure 2.

The evaluation of a given solution is performed by gathering key metrics, such as cache hit rate, from the running system OCs for analysis by the LEC. The LEC also "reports" to the higher level (Meta and Master) Evolutionary Controllers which are in turn responsible for optimising their own utility functions.

3.2 Constraining the Search

Exploring different configurations of operational controllers involves combining elements of the CDN such as the number of origin servers to employ, which directors to employ, whether or not to use connection pooling and so on... Not every possible configuration is necessarily a sensible or viable one (e.g. it only makes sense to explore the director configuration if there exist at least two origin servers), and given the cost involved in evaluating solutions (see section 3.3), penalising an ill-conceived configuration is not an option. Varnish offers many tunable parameters and operational strategies, many of which may have dependencies between each other, making the likelihood of generating invalid configurations high.

The approach taken is to use a grammar to define the searchable state-space in the same way that a grammar within GP might define the permissible combination of functions, operator and operands. Similar approaches have been adopted in other domains (e.g. [2]).

3.3 Evaluating Solutions

One of the challenges in implementing this evolutionary system is evaluating candidate solutions on a live, real-time CDN. Having access to a simulation or algorithm to be able to quickly and efficiently assign a fitness value to a candidate solution is a key component in building a powerful evolutionary system. At present, neither is available for this project which constrains the range of algorithms which may be explored to those such as (1+1)EA. Exploring the idea of a Digital Twin to create a digital replica of a running system would allow for easier evaluation, as well as expanding the number of evolution strategies available.

Adding further to the challenge of evaluation is the configuration of the Varnish CDN setup. Varnish makes use of a domain specific language known as the Varnish Configuration Language (VCL). VCL files are loaded onto Varnish Cache instances which dictates their behaviour and composition. However, this means that OCs cannot be manipulated (or evaluated) independently as only one VCL file exists per cache, and the entire cache configuration needs to be specified prior to evaluation. In order to create VCL files, a *VCLWriter* is used (see Figure 2). *VCLWriter* is responsible for translating the output of OCs into a legal VCL file which is passed onto the *VCLDeployer* which deploys the VCL file onto a target Varnish Cache which then runs with the new configuration.

4 CONCLUSIONS

This paper has reported on three key challenges we have encountered when trying to introduce autonomous evolution into a large-scale CDN. Future work will be aimed at evaluating a range of evolutionary strategies as well as the development of a digital twin to enable the rapid evaluation of multiple solutions.

REFERENCES

[1] John Dilley, Bruce Maggs, Jay Parikh, Harald Prokop, Ramesh Sitaraman, and Bill Weihl. 2002. Globally distributed content delivery. *IEEE Internet Computing* 6, 5 (2002), 50–58.

[2] José María Font, Roberto Izquierdo, Daniel Manrique, and Julian Togelius. 2016. Constrained Level Generation Through Grammar-Based Evolutionary Algorithms. In *Applications of Evolutionary Computation - EvoApplications 2016 (Lecture Notes in Computer Science, Vol. 9597)*, Giovanni Squillero and Paolo Burelli (Eds.). Springer, 558–573. https://doi.org/10.1007/978-3-319-31204-0_36

[3] Paul Harvey, Alexandru Tatar, Pierre Imai, Leon Wong, and Laurent Bringuier. 2021. Evolutionary Autonomous Networks. *Journal of ICT Standardization* (2021), 201–228.