INTERNATIONAL
CARPATHIAN CONTROL CONFERENCE
ICCC'2010
Eger, Hungary
May 26-29, 2010

MODEL-BASED CONTROL OF IT INFRASTRUCTURES

G. J. PALJAK[1], I. SZOMBATH[1], I. KOCSIS[1], T. KOVÁCSHÁZY[1], A. PATARICZA[1]
[1] Budapest University of Technology and Economics, Budapest, Hungary,
{paljak, szombath, ikocsis, khazy, pataric}@mit.bme.hu

Abstract Rapidly reconfigurable cloud computing data centers apply a feedback loop scheme to meet service level objectives. We describe our approach to use *sensor selection* to reduce over-instrumentation for monitoring and to identify *performance containment regions* by discovery for more faithful modelling and control in resource-shared systems.

Key words: IT infrastructure, monitoring, over-instrumentation, structure discovery

1 Introduction

The IT infrastructure is a system of interconnected hardware and software components that cooperate to provide business service. IT system management has to assure the high level of dependability and performability of such systems in order to meet client requirements, e.g. service level objectives (SLOs) of quality of service (QoS) metrics, while ensuring efficiency in resource utilization to reduce costs.

The prevailing industrial approach is still dominated by the former age of manual system administration, which is weakly automatized, often composed of hoc processes, strongly dependent on human administrators.

Modern data centers consist of a multitude of heterogeneous components with dense, complex interdependencies. This is especially true with the spread of dynamic architectures like cloud computing and virtualization. In cloud computing data center applications are provided as standardized offerings to end users over a dynamically reconfigurable IT infrastructure. Virtualization enables the consolidation of many under-utilized (virtual) machines on a single or some physical machines, and also the rapid reconfiguration of the virtual to physical machine mapping upon change of workload or presence of faults.

As the components in the network and the interconnections among them grow exponentially, traditional ad hoc system management is insufficient. Systematic system management has to be established that applies a supervisory control scheme for guaranteeing a high level of service by (re-)allocating redundant resources in the system to critical functions (Fig. 1.).

2 System management as a control loop

System management has to assure an

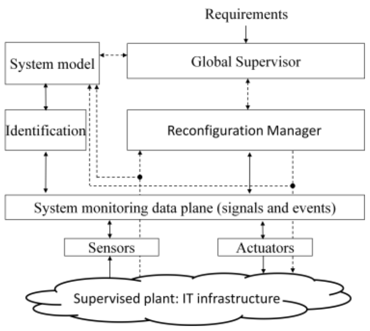

Fig 1. Supervisory control for IT management
]

appropriate reaction to changes in the environment and in the system itself originating both in evaluation and faults.

There are similarities between control theory and system management, however, there are differences as well. Sensor configuration design and measurement-based identification are well-known in both cases, but IT monitoring sensors and the frequent structural changes require new approaches.

Traditional control typically relies on few input signals (measured by sensors) as (a.) systems are well characterized by even this small set of sensors, (b.) large amounts of human effort is invested in designing sensor configurations, describing, modelling input signals, (c.) additional sensors are often problematic or expensive to install on a plant. In contrast, computer systems have thousands of measureable attributes and it is easy to install sensors (performance counters). Additionally, sensors are typically pieces of software themselves; they consume system resources, often non-negligible amounts. Thus over-instrumentation can severely degrade performance.

A major difference is that the system state contains the structure of the system, and even this can change through time. These types of changes are not common in traditional control.

The relations among internal attributes are generally non-linear, the presence of saturation is common. Although, segments with linear approximation models can be found, examples, methodologies can be found in [1].

2.1 CMDB, the IT Infrastructure Map

Configuration Management Database (CMDB) is a datastore where information about the infrastructure components and their mutual relationships are stored. In this way the CMDB stores the map of the infrastructure.

Efficient feedback in system management requires information both on the structure and on the state of the IT infrastructure. The state of the system is observed with monitoring applications. (Fig. 2) Obtaining information on the structure of the system is more difficult, there are no widely adopted approaches for IT infrastructure discovery and tracking applications. The discovery process identifies software and hardware assets, and its non-volatile parameters like configuration parameters. More advanced approaches [2, 3] are even capable of discovering limited (containing) hierarchy and interdependencies of components, for example a web server uses a database or an application runs on a particular hardware.

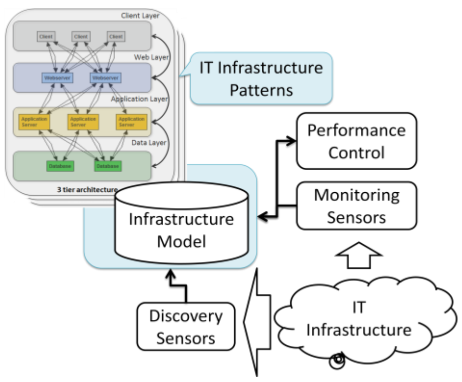

Fig. 2 Architecture of structure discovery

However there is no mature methodology for selecting the structural changes that are significant for controlling a selected set of quality of service metrics. In turn, this significantly hinders the development of effective control strategies.

We established a framework that is capable of tracking structural changes of the IT infrastructure and may help arrange the components into hierarchy.

2.2 Known approaches on IT management control loops

Feedback control has been successfully used for managing IT systems to meet SLOs [1]. In [6] a linear multi input multi output (MIMO) model is considered with a proportional integral (PI) controller. The control input and measured system output selection is by human expertise. Authors of [5] use control models to solve hot spot contention in network-level congestion control. In [4] a control is designed to meet delay requirements while saving energy by adaptively scaling frequency/voltage in a simulated portable device. Scaling and handling reconfiguration (especially structural changes) in the system are generally out of the scope of these approaches.

3 Our approach

For coarse-grained, predictive performance control, we propose utilizing the supervisory and management tooling that is usually already present in enterprise systems.

We model the performability state of distributed services qualitatively by the three classes, identified in [9], 'normal', 'degrading' and 'saturated'. The classes are defined by an implicit categorization of the load on the system via setting up respective QoS intervals. We predict the performability state of services using the monitoring data to enable setting up reaction rules that, on one hand capture the business-level requirements, and on the other hand collectively form a control algorithm.

System monitoring tools are usually deployed so that each and every performance attribute that has instrumentation is collected. Overmonitoring is not only a problem due to the inefficient utilization of system resources, but is also a significant factor for system administrators setting up ad-hoc and unmaintainably complex alerting configurations. We use variable selection (the minimum-Redundancy-Maximum-Relevancy, mRMR, algorithm [10], in detail description in [9]) to reduce instrumentation and to drive neural network-based predictions of performability states.

Data centers are resource shared environments: multiple services may use the same network, hardware and software resources. Determining the cross-service performance effects of resource sharing is generally a very hard problem; for practical purposes individual control instances should be designed for system partitions that are independent from resource utilization and contention point of view.[8] While this may mean that performance control we have to aggregate multiple services (e.g. two services using the same database server). Modern data center technologies – most importantly resource capping mechanisms in host and network virtualizations – allow partitioning the system from the resource usage point of view. For instance, two virtual machines using the same host, but resource separated by appropriate CPU allowance capping will be coarsely independent, free of CPU-related usage side effects.

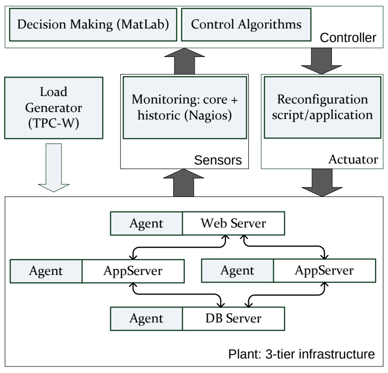

Fig. 3 Three-tier pilot infrastructure

However, identifying these *performance containment partitions* is clearly in practically intractable problem for a large, distributed and dynamic environment. As such, we propose implementing partition identification on top of CMDBs. Largely for the same reasons why fault containment regions identification also needs a CMDB in such environments.

Utilizing CMDBs for this purpose has some open questions: most importantly, what CMDB relationship types are needed and how a CMDB model space should be transformed to performance containment regions in general.

4 Test infrastructure

An experimental application infrastructure (Fig. 3) was set up to emulate a scaled-down datacenter with a monitoring and control framework integrated. The three-tier server system is tested with the TPC-W benchmark [7] workload. The application servers are in a load-balanced, fault-tolerant, runtime reconfigurable cluster. Monitoring data gathered by sensor agents of a widely-used monitoring solution (Nagios) are processed in Matlab.

The infrastructure and the monitoring and control framework are installed on virtual machines that are deployed on two separate physical machines. This setup avoids measurement bias (that might be a consequence of co-location on a single machine) by physical separation, reckoning with performance containment partitions.

4 Experimental results

In our experiments, we started with a naive approach of monitoring and measured more than 130 performances attributes (typical out-of-the-box monitoring scenario). Then, we reduced the number of metrics to the 8 most relevant to the range of a system level QoS metric, throughput, with the mRMR algorithm (similarly to [9]) to decrease monitoring overhead. We used 75% of the measurement data in a 16 hour long experiment to train neural networks to predict the performability state of the system. The rest of the data is used for validation. A typical measurement-prediction scenario can be seen on Fig. 4 (states are: 0 – 'linear', 1 – 'degrading, 2 – 'saturated' state). We have achieved 84-86% prediction accuracy for 1, 3, 5 minute prediction horizons; this empirical evidence affirms our approach and encourages further research.

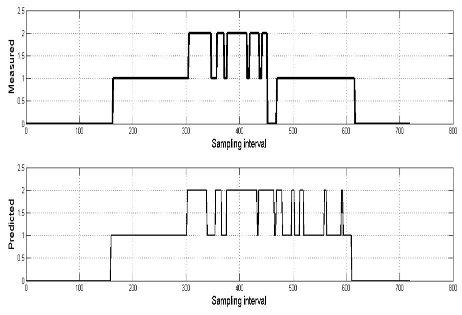

Fig.4. Performability state prediction

References

[1.] J. Hellerstein, Y. Diao, S. Parekh, and D. Tilbury, *Feedback control of computing systems,* IEEE, 2004.

[2.] A. Kind, D. Gantenbein, H. Etoh, *Relationship Discovery with NetFlow to Enable Business-Driven IT Management*, Workshop on Business-driven IT Management, IEEE, 2006.

[3.] Y. Chen, M. Zhang, Z. Morley Mao, P. Bahl, *Automating Network Application Dependency Discovery: Experiences, Limitations, and New Solutions*, OSDI, pg. 117-130, 2008.

[4.] Z. Lu, J. Hein, M. Humphrey, M. Stan, J. Lach, and K. Skadron, *Control-theoretic dynamic frequency and voltage scaling for multimedia workloads,* In Proc 2002 Compilers, Architecture, and Synthesis for Embedded Systems, 2002, p. 156–163.

[5.] G. Pfister, M. Gusat, W. Denzel, D. Craddock, N. Ni, W. Rooney, T. Engbersen, R. Luijten, R. Krishnamurthy, and J. Duato, *Solving hot spot contention using infiniband architecture congestion control,* HP-IPC 2005, 2005, pp. 41-44.

[6.] N. Gandhi, D. Tilbury, Y. Diao, J. Hellerstein, and S. Parekh, *MIMO control of an Apache web server: modeling and controller design*, Proceedings of the 2002 American Control Conference , pp. 4922-4927.

[7.] TPC-W official page: http://www.tpc.org/tpcw/default.asp

[8.] O. Tickoo, R.Iyer.,R. Illikkal,D. Newell, D.. *Modeling virtual machine performance: challenges and approaches,* SIGMETRICS Perform. Eval. Rev. 37, pp.55-60, 2010

[9.] G.J. Paljak, I. Kocsis, Z. Egel, D. Toth, A. Pataricza, *Sensor Selection for IT Infrastructure Monitoring,* In: Autonomic Computing and Communications Systems, Springer, 2010, pp. 130-143.

[10.] H. Peng, F. Long, C. Ding, *Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol . 27, pp. 1226-1238, 2005