Support for Automatic Diagnosis and Dynamic Conf guration of Scalable Storage Systems

Zsolt Németh[1], Michail D. Flouris[2], Renaud Lachaize[3], and Angelos Bilas[3]

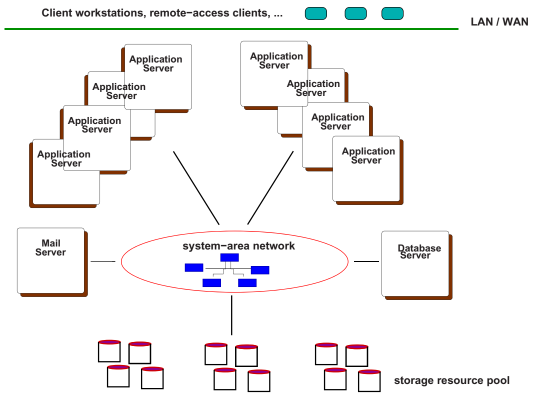
[1] MTA SZTAKI Computer and Automation Research Institute
P.O. Box 63, Budapest, H-1518, Hungary
zsnemeth@sztaki.hu
[2] Department of Computer Science, University of Toronto,
Toronto, Ontario M5S 3G4, Canada
[3] Institute of Computer Science (ICS), Foundation for Research and Technology - Hellas
P.O.Box 1385, Heraklion, GR 71110, Greece
{flouris,rlachaiz,bilas}@ics.forth.gr

Abstract. Distributed storage systems are expected to serve a broad spectrum of applications, satisfying various requirements with respect to capacity, speed, reliability, security at low cost. Virtualization techniques allow ˜exible con° guration of storage systems in order to meet resource constraints and application requirements. Violin provides block level virtualization that enables the extension of storage with new mechanisms and combining them to create modular hierarchies. Creating and maintaining such virtualization hierarchies however, is a complex task where a human system administrator is the most expensive and less ef° cient element. We introduced Conductor, an automated support system that tries to grasp human expertise with declarative rules that are applied to storage management. So far the initial, static con° guration capabilities of Conductor have been elaborated. Static features however, are not suf° cient for practical purposes as the storage system evolves, i.e. requirements, workloads, access patterns may change in time. This paper presents work in progress that is aimed at extending Conductor with supporting dynamic features. We introduce the concepts of global and directed recon° gurations and discuss their potential strengths and weaknesses.

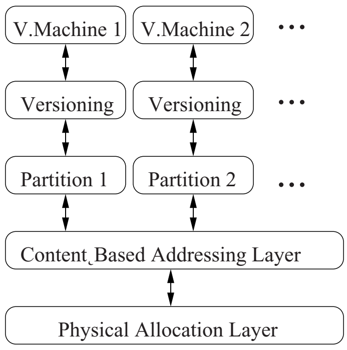
Keywords: distributed storage management, virtualization, rule based system.

1 Introduction

As the volume of digital data increases, scalable storage systems provide a means of consolidating all storage in a single system to improve cost-ef° ciency (Figure 1a). For this reason, storage system architectures are undergoing a transition from directly- to network-attached. This new architecture offers potential for ˜exible con° guration of storage systems to better match application needs and thus improve their performance. This is an important concern because distinct application domains have very diverse storage requirements; Scienti° c computation, data mining, e-mail serving, e-commerce, search engines, operating system (OS) image serving or data archival impose different



(a) Networked storage architecture



(b) Virtual hierarchy example

Fig. 1. Generic networked storage organization (a) and example virtual hierarchy to consolidate storage for virtual machines

tradeoffs in terms of I/O throughput, latency, reliability, capacity, high-availability, security, data sharing and consistency.

Violin [3] is a kernel-level framework for building and combining virtualization functions at the *block* level. Violin targets commodity storage nodes and replaces the current block-level I/O stack with an improved I/O hierarchy that allows for (i) easy extension of the storage hierarchy with new mechanisms and (ii) ˜ exible combining of these mechanisms to create modular hierarchies with rich semantics. As an example, Figure 1b shows a virtual hierarchy that provides a private virtual disk to each of many virtual machines running on a single system [4]. This system uses partitioning, versioning, and content addressable storage layers to provide the illusion of private disks on top of the same physical storage. More scenarios of advanced virtualization semantics are discussed in [4].

We proposed Conductor [2], a rule-based expert system that is aimed at providing support for con° guring and maintaining virtual storage hierarchies in scalable storage systems, such as Violin. Currently, this task relies entirely or mostly on the expertise and intuition of human system administrators. Moreover, most con° guration activities are usually complex, not exact, and thus, hard to formalize. Conceptually, Conductor maintains a knowledge base about the storage system as facts, e.g. devices, properties, measured values, structures, and expertise expressed as rules, e.g. how the characteristics of a disk are changed if striped or what are the symptoms of a faulty disk. Based on the facts and rules Conductor will be able to infer new information that may indicate symptoms of problems or may trigger corrective recon° guration actions.

Storage requirements of applications can be divided in two categories: (a) Statically satis°ab le requirements, such as capacity, archival capability, fault tolerance level, encryption, and compression. (b) Dynamically satis° able requirements that refer mostly to performance characteristics, such as throughput and response time, albeit, some static requirements may also change over time for a given application.

In its current status, as presented in our previous work [2], Conductor is able to deal with static (initial) system con° guration. The focus of the work so far has been to suggest optimal con° gurations that fully satisfy functional (static) requirements, but only approximate performance (dynamic) requirements based on estimated performance values for system components. The performance of a storage element depends both on its physical characteristics as well as the speci° c application workload. The latter is usually only approximately known at system con° guration time. Therefore, initial (static) system con° guration usually relies on estimated values for dynamic (performance) characteristics. This is also what happens in Conductor; For instance, when a system needs to provide a virtual volume that offers a certain level of I/O throughput, Conductor relies on estimated values for the throughput of physical devices and heuristics to estimate the throughput of the ° nal virtual device. In essence, Conductor currently addresses two challenges:

– It captures human expertise in the form of rules of a production system. They represent "rules of thumb" that a human administrator would follow, e.g. "to achieve a certain level of throughput stripe a virtual volume over a number of devices". However, to grasp real human expertise and introduce sophisticated rules beyond elementary ones currently implemented, this aspect of Conductor needs to be further investigated.
– It reduces the complexity of searching the con°guration space. In this direction, Conductor uses heuristics and tries to reduce search complexity without compromising the quality of the generated solutions signi°can tly.

In this paper, we focus on how to augment Conductor in order to satisfy dynamic requirements, as both workload and system characteristics evolve over time.

Satisfying dynamic requirements requires continuous monitoring of a storage system to detect whether certain goals are being violated. Monitoring is system speci°c and is usually possible by instrumenting the I/O path at user- or kernel-level. Monitored data are inserted into Conductor's knowledge base in form of facts and subsequently Conductor is able to detect if certain (dynamic) requirements are not met by simply comparing the monitored information to the original speci° cations; For instance, if throughput of a speci° c volume drops below a minimum threshold during high traf° c conditions, this may imply that the system is not able to satisfy application requests at the agreed rate. Whenever Conductor detects a discrepancy from the original speci° cation it triggers corrective actions that will recon°gure the system. Now, we discuss two main alternative approaches to dynamic recon° guration: (a) global dynamic recon° guration and (b) directed dynamic recon° guration.

2 Global Reconfi uration

One approach to deal with dynamic features is to trigger a full system recon° guration when problems are detected. This procedure resembles static con° guration: a new virtual hierarchy is built from scratch, however, using actual, measured values as opposed to the estimated values used in static system con° guration. For instance, if throughput to a speci° c physical disk is measured to be half of the estimated throughput, then this measured value may lead to more realistic con° guration. This scheme can be further re°n ed by establishing certain statistical properties of measured values or relationships between them. As an example it may be projected that encrypted volumes have x percent higher latency where x is established from actual measurements. Hence, not only can measured values replace estimated ones, but also actual experience can re°ne the way con° guration is realized by updating the knowledge base in Conductor. In this sense, measured values also serve as re° nements to the existing experience over a longer period; certain trends, relationships between performance and workload, further details of projecting the performance can be established and incorporated into both static and dynamic (re)con° guration.

Global reconfiguration is a natural extension of static con° guration and re-uses existing mechanisms in Conductor. While initial con°guration may not meet performance requirements due to the use of estimated values for performance characteristics, recon° guration of the system using feedback from the actual system can narrow the gap between required and achieved values. However, this approach has several potential disadvantages.

A main issue is the overhead that a global recon° guration incurs. Our previous work [2] shows that this is an extensive process. First, con° guration itself is an exhaustive search and even though various search strategies have been investigated to reduce complexity by several orders of magnitude, it may still exhibit an exponential behavior. Moreover, recon° guration affects the entire storage hierarchy independently of the type or focus of the problem, which may not scale in large storage systems. Finally, another potential weakness of global recon° guration is that although it uses more realistic actual performance values, it omits workload information. Workload behavior may depend on the structure of the hierarchy itself. Thus, rebuilding the full hierarchy from scratch may result in different workload behavior and, as a consequence, reduce the usefulness of the measured values.

3 Directed Reconfigurat on

Instead of triggering a global system recon° guration, an alternative approach is to ° rst detect the type of performance problem as accurately as possible as well as the location where it occurs in a storage hierarchy and then solve it by *directed, local* recon° guration. This, less intrusive approach requires detecting the origins of discrepancies in dynamic characteristics. Today, this *diagnostic* procedure is the task of experienced human operators that understand both application requirements as well as symptoms of speci° c performance problems. Automating this procedure is essential for improving the cost-ef°cienc y of large scale storage systems.

Directed recon° guration relies on the inference mechanism of production systems [5] that is especially appropriate for diagnostic purposes. If the expertise of a human operator is expressed as rules in the production system, then measured facts (monitored values) about system components can be used to infer (diagnose) the location of a problem.

The location where a problem is ° rst detected is not necessarily the origin of the problem. For this reason, diagnosis involves multiple, recursive steps, where inferred information at each step may lead to further decomposing the system to simpler components. For instance, if the bandwidth of a volume is less than expected, diagnosis needs to examine which of the constituent virtual or physical devices of the volume may be responsible. This recursive procedure, essentially follows the structure of virtual volumes, as they are composed out of (possibly multiple layers of) physical devices, storage nodes, and network paths. When a problem is localized with diagnosis, the actions that will be taken by directed recon° guration represent another form of human expertise and are empirical, inexact, and hard to formalize.

Directed recon°guration may lead to better decisions compared to global recon° guration, since diagnosis tries to preserve as much as possible the existing structure. This allows for selecting the most promising recon° guration actions in a given situation and can lead to better con° gurations in fewer steps, avoiding a costly (and potentially more disruptive) global recon° guration that has to try a plethora of possibilities. For example, if there is an indication that 10% more bandwidth is required in a virtual volume, directed recon° guration may suggest immediately an upgrade of the volume from 2- to 3-way striping, instead of trying all possible con° gurations using measured performance values.

The main drawback of directed recon°gura tion is that in several cases it may not be easy to ° nd the exact scope and cause of a problem. Even human system administrators sometimes can do little more than an intelligent guess – this type of experience can hardly be formalized. It is thus likely that only a subset of the potential problems and their symptoms will be formalized as rules. Nevertheless, we anticipate that signi°cant classes of performance problems can be detected by this method and addressed by ef° cient system recon° guration.

4 Related Work

Due to the overall complexity of administering storage systems, the application of abstract control and intelligent methods, have been proposed in recent work. While there are some similarities with our work in certain details, none of them address the issue of con° guring and maintaining virtual storage hierarchies.

Polus [7] aims at mapping high level QoS goals to low level storage actions by introducing learning and reasoning capabilities. The system starts with a basic knowledge of a system administrator expressed as "rules of thumb" and it can establish quantitative relationships between actions taken and their observed effects to performance by monitoring and learning.

Ergastulum [1] is aimed at supporting the con° guration of storage systems with reducing the search complexity of possible design decisions by utilizing heuristics with randomization and backtracking.

A novel approach presented in [6] tries to predict the effect of certain actions and helps with making decisions at data distribution. It establishes a set of *What... if...* statements where the hypothetical effect (what part) of a certain circumstance (if part) is stored. These relations are obtained by statistical, analytical or simulation methods.

5 Conclusions

Our goal in this work is to examine how we can extend the existing static features of Conductor [2] to automatically con° gure large scale storage systems so that they satisfy application requirements for dynamic system characteristics. We introduce two potential approaches: The ° rst, global recon° guration, is a direct extension of Conductor and can be implemented in a straightforward manner. The second one, directed recon° guration aims at capturing further human expertise when managing large storage systems. One of the main challenges here is introducing appropriate diagnostic rules in the production system.

Global and directed recon°guration can also be seen as complementary to each other: Global recon° guration uses measured data but omits structural information. Its effect is global and it is most useful when problematic spots cannot be identi° ed either because they are related to the entire structure with no speci°c focus or appear too frequently or simply cannot be diagnosed. On the other hand, directed recon° guration takes into account measured data *and* structural information and tries to locate the problematic spot and the possible causes as precisely as possible. It is more appropriate for "local" problems in the system structure, such as performance hot-spots.

Finally, we are currently implementing the two approaches. This requires addressing the following challenges: (a) Capturing human expertise in the form of rules for diagnosis purposes can happen at various levels of detail. (b) Although detecting a certain problem is easy, it is hard to decide if action must be taken or if the problem is temporary, can be tolerated and should thus, be ignored. (c) Improving the management system by extending the knowledge base gradually with new rules as more experience is acquired with new applications and new system components. (d) Performing experiments with realistic setups that re˜ ect situations encountered in real life.

Overall, we believe that rule-based expert systems, such as Conductor, tuned to the needs of storage applications, can offer signi°cant help in managing large scale storage systems and improving their cost-ef° ciency.

References

1. E. Anderson, M.Kallahalla, S. Spence, R. Swaminathan, Q.Wang. Ergastulum: Quickly Finding Near-Optimal Storage System Designs. HP Laboratories SSP technical report HPL-SSP-2001-05 (2002)
2. Zs. Németh, A. Bilas, M.D. Flouris, R. Lachaize: Conductor: An Intelligent Con° guration Framework for Storage Area Networks. Book on Knowledge and Data Management in Grids, CoreGRID series, Springer Verlag, 3, 2006
3. M.D.Flouris, A. Bilas: Violin: A Framework for extensible Block-level Storage. 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST 2005) April 2005, Monterey, CA, USA. IEEE Computer Society 2005.
4. M. Flouris, R. Lachaize, and A. Bilas. Violin: a Framework for Extensible Block-Level Storage. Book on Knowledge and Data Management in Grids, CoreGRID series, Springer Verlag, 3, 2006

5. M.Klein, L.B.Methile: Expert systems: A Decision Support Approach. Addison-Wesley, 1990.
6. E. Thereska, M. Abd-El-Malek, J. J. Wylie, D. Narayan an, G. R. Ganger. Informed data distribution selection in a self-predicting stor age system. Proc. of the International Conference on Autonomic Computing, ICAC-0 6, Dublin, Ireland, June 2006.
7. S. Uttamchandani, K. Voruganti, S. Srinivasan, J. Palmer, D. Pease. Polus: Growing Storage QoS Management Beyond a "Four-year Old Kid". USENIX FAST '04 Conference on File and Storage Technologies, March 2004, San Francisco, CA, USA.