# A Probabilistic Approach to Distributed System Management

Randy N. Schauer
University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering
Baltimore, MD 21250
+1 410 227 0580
schauer1@umbc.edu

Anupam Joshi
University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering
Baltimore, MD 21250
+1 410 455 2590
joshi@cs.umbc.edu

## ABSTRACT
Large-scale distributed systems are playing an increasing role in computational research, production operations, information processing, and application hosting. The continuous management of such systems is a critical consideration when focusing on reliability, availability, and security. As the number of commodity components within these systems continue to grow, it becomes increasingly difficult to track the multitude of parameters required to ensure optimal performance from the system, especially in those systems that have been built through expansion and not as an initial purchase of identical nodes. In this paper, we discuss the use of statistical inference, specifically Markov Logic Networks, in a distributed multi-agent system to provide the most effective means of managing these parameters. We showcase an architecture that provides services to manage a system's configuration throughout its life-cycle, and is capable of resolving differences after identifying potential mis-configurations using conflict discovery and resolution modules.

## Categories and Subject Descriptors
G.3 [**Probability and Statistics**]: *markov processes, probabilistic algorithms, statistical computing.*

I.2.3 [**Artificial Intelligence**]: Deduction and Theorem Proving – *deduction, inference engines, uncertainty.*

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Foundations and Methods – *predicate logic, temporal logic.*

K.6.4 [**Management of Computing and Information Systems**]: System Management – *centralization/decentralization.*

## General Terms
Algorithms, Management, Reliability, and Experimentation.

## Keywords
Configuration, Distributed, Management, Probability.

## 1. INTRODUCTION
Data centers, particularly those providing high-end computing services, continue to provide more power and resources while simultaneously implementing green initiatives and virtualized hosts to leverage appropriate economies of scale. Managing this complex nest of systems that have varying requirements, configurations, and access rights in an efficient manner requires a cadre of highly-trained system administrators. To alleviate increasing workloads placed on the staff, freeing them up for more cerebral tasks, the goal of automated system management must be achieved.

The purpose of our research is to provide a true distributed approach to system management, with the means to administer ever-growing systems, both in physical size and density, without increasing the administration staff. To reach this goal, we must investigate the use of autonomic principles on each node of a large distributed system. The primary task to achieve this is twofold. First, the system must be able to manage aspects of its configuration without using a central image master, relying only on the knowledge of its peers. Second, the system must be able to understand and evaluate its operating environment to identify issues before they become catastrophic problems. In each of these scenarios, the system should rely on itself to provide the necessary means for self-management using a combination of historical analysis and information relayed from nodes currently operating in a similar configuration.

Our approach provides a flexible solution that can be adapted to different types of distributed systems in order to remove the centralized knowledge base as both a single point of failure and a network bottleneck. This allows us to implement an intelligent technique for evaluating potential changes needed to keep the system in a consistent state. The proposition is that through dynamically adapting to the current state of the system, a more consistent, better tuned environment for the end users will be provided without compromising functionality or availability.

## 2. ARCHITECTURE
Our current architecture consists of four functions: Data Gathering, Probabilistic Inference, Conflict Discovery, and Resolution. Each of these functions provides a necessary service, with the Probabilistic Inference function being the most complex and important in determining if the system configuration is accurate or not.

### 2.1 Data Gathering
The Data Gathering module exists on each node in a distributed system, collecting information about parameters, properties and


*ICAC'10*, June 7–11, 2010, Washington, DC, USA.
ACM 978-1-4503-0074-2/10/06.

settings to allow for a complete analysis of system state. Once the data is gathered, it is placed into the appropriate format for the Probabilistic Inference module, and passed off to be consolidated into a single input deck for analysis.

## 2.2 Probabilistic Inference

The Probabilistic Inference model verifies that every system node is approximately identical to other related nodes. By performing comparisons on multiple nodes, statistical relational learning methods will be used to determine the optimal configuration and make adjustments as necessary. Due to the infinite possible optimal configurations for differing environments, a statistical relational learning method is the preferred inference mechanism, specifically Markov Logic Networks (MLNs) [1].

MLNs provide a first-order predicate knowledge base with a weight applied to each formula. The logic being used allows for an initial set of conditions that capture the rules needed to make informed decisions in the inference phase. The softening constraints that a MLN provides as compared to strict first-order logic allows for an expanded view of options when changes are being considered [3]. The probabilistic inference performed within MLNs provides an effective means to distribute the calculation of the correct configuration, while still ensuring that nodes will sustain a secure operational baseline for their users.

As systems age, node configurations can and do change to accommodate updated software, libraries, patches and hardware replacements. Once a potential issue is discovered, nodes will use the probabilistic inference engine to form an agreement on which configuration value is correct. To ensure each node is sending an accurate description of itself for each set of comparisons, time comparisons will occur at multiple stages to determine if they were gathered at the "same" time across the cluster. Latency is a concern for any type of distributed system, but can pose additional issues in situations such as this.

## 2.3 Conflict Discovery and Resolution

The Conflict Discovery module performs the final analysis of the Probabilistic Inference module's results. At this stage, the data is reviewed to determine what, if any, corrective actions need to be taken on various nodes. During the course of analysis, corrective actions are grouped by node, and communicated back to the appropriate Resolution module.

The Resolution module exists on each node, and is responsible for performing the actual corrective actions as dictated by the Probabilistic Inference and Conflict Discovery modules. Depending on the results of the analysis phase, a node may or may not have to wait for idle time to correct itself to prevent interference with user jobs. The Resolution module verifies that the conflict still exists prior to taking corrective action to ensure the planned action still corrects the identified problem.

# 3. INFERENCE MODEL APPROACH

Our inference models contain a number of setting, parameter and environment values that require consistency across a large distributed system. A statistical approach to solving these issues as they arise can take all the known factors into account and weight them to minimize uncertainty and determine the most valid option.

The MLN probabilistic inference calculations are being performed using the Alchemy System for Statistical and Relational Artificial Intelligence [2]. In this system, there are two stages which must be executed in order to perform the analysis. The first stage is weight learning with the MLN formulas and training data knowledge base. The system allows for both generative and discriminative learning, providing new weighted MLN data that will be used for the inference model.

The second stage takes the weighted MLN output and uses that along with the evidence knowledge base to perform an inference calculation on the network. The system allows for various types of inference, including Gibbs sampling, MC-SAT, simulated tempering, and Maximum A Posteriori (MAP). The output of each inference calculation is the statistical probability that an evidence predicate is not true.

The goal of this set of inference models is to form a distributed, intelligent system integrity validation product that ensures an optimal configuration while simultaneously watching for attempted infiltrations. Performing configuration control in this manner will greatly improve the productivity of staff responsible for maintenance on both large and small distributed systems.

# 4. CONCLUSION

The management of distributed systems such as large computational clusters is a non-trivial task. As systems continue to grow in size and both the quantity and quality of services offered, there will continue to be adoption of these systems in industries outside what has been seen as the norm in the past. The ability for all types of distributed systems to diagnose and recover from performance and configuration issues without resorting to a centralized knowledge base is the next great stride in allowing systems to self-manage their reliability and stability in this high-end community.

The prototype system we have built utilizes a novel approach to performing a probabilistic model-based diagnosis. We are removing single points of failure for diagnosis and allowing the system to correct different types of issues itself. This prototype system and the results as a means to incorporate industry best practices along with some of the latest research being done in the area of statistical relational learning and autonomic computing.

# 5. REFERENCES


[1] L. Getoor and B. Taskar, Introduction to Statistical Relational Learning (The MIT Press, Cambridge, MA, 2007).

[2] S. Kok, P. Singla, M. Richardson, and P. Domingos. The Alchemy system for statistical relational AI. Technical Report, Department of Computer Science and Engineering, University of Washington, http://www.cs.washington.edu/ai/alchemy, 2005.

[3] M. Richardson, and P. Domingos. Markov Logic Networks. *Machine Learning*, 62, 107-136, 2006.