A data mining based approach to reliable distributed systems

Michael Mock and Dennis Wegener
Fraunhofer IAIS
Schloss Birlinghoven
53754 St. Augustin, Germany
{michael.mock,dennis.wegener}@iais.fraunhofer.de

***Abstract*—The purpose of this paper is to open a novel research perspective on reliable distributed systems. The underlying hypothesis is that dynamic models of distributed systems can be established by the use of data mining techniques being applied to data gathered in observing the distributed systems. Feeding observations in an on-line monitoring process into such a model allows predicting upcoming reliability and performance problems, thus enabling the user or the system to take preventive measures for increased reliability or performance. We present the general approach and elaborate a concrete scenario of applying this approach in the field of distributed data mining algorithms.**

Keywords-data mining; distributed systems; reliability; performance

I. INTRODUCTION

Reliability of distributed no longer can be assured by static design because distributed systems are increasingly large, heterogeneous and dynamic. On the one side, large-scale computing grids, clouds and clusters provide computing and data resources with hundreds or even up to thousands of nodes. On the other side, the development of small, embedded potentially mobile devices with wireless communication facilities allows for the creation of wireless networks with hundreds of nodes. In both scenarios, reliable operation in case of faults and disturbances no longer can be achieved based on static design decisions using a static system model. The approach taken in this work is to generate dynamic system models by applying data mining techniques on data that is gathered by a controlled monitoring process during the run-time of the system. These models will allow predicting future system behavior on the basis of actual monitoring data. As a consequence, performance problems of applications in large infrastructures and failures such as loss of communication coverage can be predicted before they occur, thus allowing to take preventive measures at run-time in order to increase the overall reliability of the distributed system.

II. RELATED WORK

Closest to and also inspiring our work is the work described in [6]. The authors present a framework and experiences with predicting system behavior from past observations. Whereas they mainly focus on time series and models based on stochastic processes, we aim at investigating the use of general purpose data mining techniques such as classification and clustering for system modeling. The applicability of data mining techniques for troubleshooting distributed systems is already mentioned in [2]. The execution logs of a 250-node Condor cluster are transformed into a feature space, which is examined with standard data mining algorithms for building decision trees and decision rules. In some synthetic examples using injected faults, the authors show that it is possible to detect which jobs have failed on which type of machines and to derive rules for future job submissions. The authors conclude that their result indicates that it is promising to apply similar approaches to real-life applications. However, in real-life applications, there is often no simple distinction between failure and success as in the examined, synthetic example. Our approach aims at supporting the prediction and detection of QoS related faults, which are more complex than simple binary fault indicators. A further restriction mentioned by the authors is that their experiment only allows predicting the probability of failure for identical computations in identical system environments, a limitation which we aim to overcome by continuous monitoring, model adaption and domain specific similarity modeling.

In [15], an automated approach on performance problem localization in service-oriented systems based on Bayesian networks is presented. The approach focused on isolation of problem causing services based on end-to-end response times. The approach and the presented architecture can work with other models and also support more fine grained analysis, but in contrast to our approach the prediction of reliability and performance problems seems to be not considered.

[12] presents an approach to characterization, modeling and prediction of dynamic resource availability in a large scale grid environment. In this

approach, different classes of resource availabilities, which can vary due to policies, are identified and individually modelled. The availability is then predicted based on Bayes- and Nearest Neighbour algorithms. The approach is more focused on availability over time than on error or fault detection. Detection of resources that are available but disturb the execution, e.g. because they are slow due to mis-configuration, partially blocked by other processes or have bottlenecks w.r.t. the scenario, seems to be not supported.

In [3], a data mining based approach on fault prediction and detection on the grid is introduced. The architecture of a distributed enactment engine (ASKALON) that supports intelligent fault detection is presented. [4] gives a model based approach for application specific fault diagnosis, implemented via wrapper services that encapsulate the original services. Users add fault diagnosis to the services, on which models are build by data mining algorithms. Both approaches of using data mining techniques are similar to our approach, but based on system parameters (middleware, memory, etc.) only. Only successful and not successful executions seem to be considered, bottlenecks and performance lacks seem to be not considered.

III. General Approach

Distributed Systems today grow dynamically in terms of new applications, hardware and network components, users and workload changes. Complex interactions between the different layers of a distributed system make systems and effects of faults hard to understand such that faulty behavior and poor performance cannot always be distinguished. Our approach to reliable operation of distributed systems is based on building a dynamic model for the distributed systems from monitored system data. Similar to our work in [12], we aim at determining system parameters that are significant for predicting relevant changes such as faults, in the system. Fig. 1 depicts the overall approach.

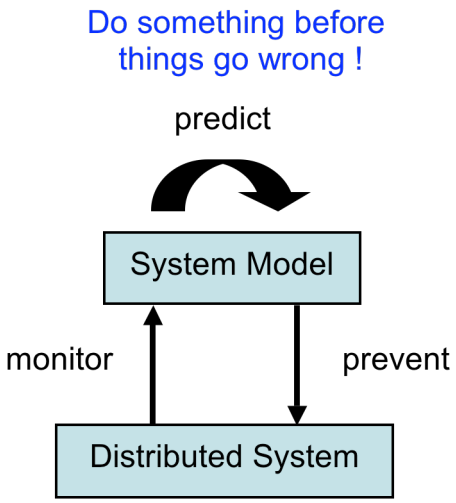

Figure 1. General Approach

In detail, we want to evaluate collected data of the system with the help of data mining techniques in order to build a model of the system. This model can be used for online prediction of the system's behavior and thus gives the opportunity to react even before faults or errors occur. The following depicts a 5-Step approach on predictive maintenance for distributed systems:

- Step 1 - **Data Gathering (logging, online monitoring)**: This step provides the basis for building the model of the system. Observational data from the operational system is gathered either in an online or an off-line setting. Using data from the real system for model building allows for much more realistic models than a-priori static models or simulations.
- Step 2 - **Model Building**: This step is based on the data mining analysis of the monitoring data. E.g., feature-selection and classification

TABLE I. Examples for the approach

	Example 1 [2]	**Example 2 [3]**	**Example 3 [4]**
Problem	A more reliable telephone router	A faster replicated file server	Replace storage replica only when needed (permanent fault)
Step 1	Analyse system and fault logs of a multiprocessor telecommunication switch (4200 System variables)	Monitor system variables of replicated file-servers	Monitor downtimes/recovery times of internet nodes
Step 2	Sem. Ops / sec + Kernel memory allocation are most relevant indicators that telephone connections requests will fail in the near future	Find out which system variables correlate under normal operation	Derive a Markov Model from the failure history
Step 3	Monitor semaphore ops and kernel mem.	Check whether file server replicas behave "correlatedly"	Use (adaptive) timeouts to detect node failures
Step 4	re-route calls to another server if a threshold is reached	replace server replica	replace node only if model decides for permanent fault
Step 5	(Adaptation of the model)	(Adaptation of the model)	(Adaptation of the model)

algorithms can be used to determine which parameters of the system are most correlated to failures or performance problems

- Step 3 - **Online-Monitoring and Prediction**: Given that a model of the system gained in Step 2 exists, selected parameters can be monitored and short term prediction on potential problems can be made on the basis of this model. Also, predictions regarding parameter settings of the distributed system become possible.
- Step 4 - **Preventive Measures**: In case that the model predicts reliability or performance problems based on the actual observations, measures might be taken to prevent problems, instance system parameters might be changed or load might be reduced
- (Step 5 - **Adaptation of the model**): As the actual system might emerge over the time, adaptations of the model itself might become necessary. Basically, the model building process taking place in Step 2 on the data from Step 1 will have to be checked against long-term changes in the actual monitored data. The model adaptation might require human interaction, but also could be foreseen automatically, for instance when re-adjusting thresholds or expected delays.

Table 1 shows three examples from current research that each can be interpreted as an application of this the 5-Step schema: Call Availability Prediction in a Telecommunication System [5] (Example 1), Replicated File Servers [8] (Example 2) and Distributed Storage Management [7] (Example 3). All these examples show that system models for complex distributed systems are built dynamically from observational data. The model is then used to predict or detect reliability and performance problems. We believe that the application of data mining algorithms will be a big help for building such models. Of course, there cannot be a single model capturing all types of distributed systems at the same time. Instead, we try to find models for specific system scenarios in which we believe that an underlying process can be identified. In the next section, we present such a scenario, which is dealing with the execution of data mining algorithms in distributed, cluster- or grid-based systems.

IV. RELIABLE DISTRIBUTED DATA MINING

In the previous section a general approach on using data mining techniques for modeling distributed systems was introduces. In this section, we focus on the application scenario of running data mining algorithms in a distributed computing environment. We present an initial system architecture and give details on the research performed in this direction.

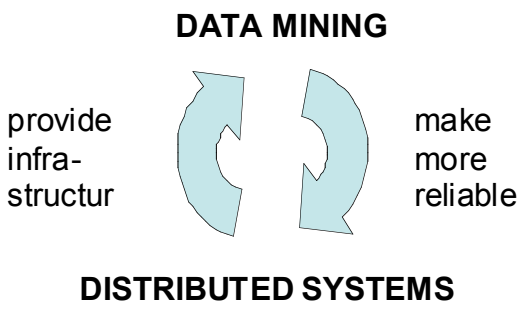

Figure 2. Data Mining and Distributed Systems

A. Data Mining and Distributed Systems

The aim of using distributed environments for data mining tasks as shown in Fig. 2 is to save time and resources in existing scenarios as well as the enabling of novel scenarios at all. Current Data Mining scenarios include large data sets (terabytes of data), distributed data sets (multiple organizations involved), privacy-sensitive data, and high computational demands. Thus, there is a clear need for state-of-the-art, rapid data analysis and for easy integration and distribution. But, depending on the point of view, a lot of questions arise when performing data mining tasks in distributed systems.

A system administrator is interested in getting to know if there are any bottlenecks in the system and if there are any components or areas in the system that cause failures. He wants to know how the system can be prevented from failures, how the system's performance and fault tolerance can be improved and in which way the system does have to adapt dynamically.

In contrast, the data miner is interested in getting to know how to parallelize or grid enable algorithms, how to parameterize them and how to organize the execution in terms of where to put the data, to make use of replication, and how many processors are needed. He wants to see if the execution makes progress (in time and in quality) and how long the computation might take (to reach a certain quality) and wants to know if an error occurred and if he has to adapt the algorithm. Fig. 3 depicts the Data Miner's point of view.

In previous work [13] we have analyzed these questions experimentally considering a specific algorithm on a specific system. The analysis has been performed "manually", i.e. by collecting log files of execution times of algorithms and keeping track of injected faults by hand. We are currently developing a distributed monitoring system for distributed data mining algorithms that will support an automated analysis. A key element is that we are not restricted to looking at the execution of the algorithms as black box, but the system allows for a fine-grained monitoring of algorithms internals. The approach and an initial architecture are presented in the next sections.

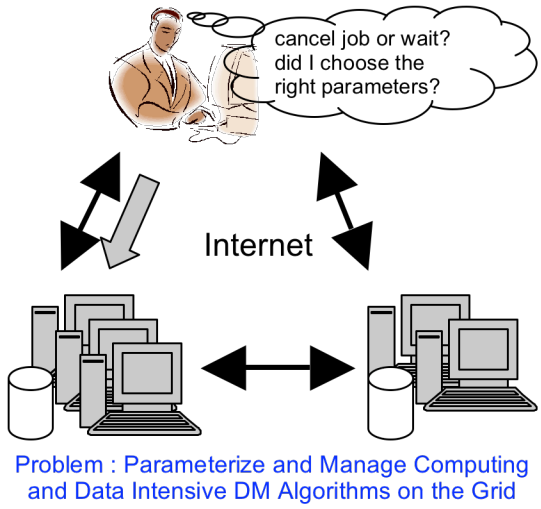

Figure 3. Data Mining as Application Scenario

B. *Adapting the approach*

The execution time and progress of a distributed data mining algorithm depends on many parameters, such as type, implementation, parallelization and parameters of the algorithm, size, pre-processing and values of the input data, and performance, load, and health-state of the targeted computing environment. The ultimate goal would be to build a model that captures all relevant parameters such that it becomes possible to predict the execution time and progress of distributed data mining algorithms. Such a prediction could be used in several manners: a data miner could compare the estimated time and progress with the actual progress of ongoing executions in an online-monitoring setting, such that irregularities can be detected and respective measures can be taken, for instance, canceling and re-submitting the computing job to different machines. Also, changes in the reliability and performance of different target computing environments can be detected, even in an off-line setting. Adapting the presented approach to standard data mining scenarios leads to the following five steps:

- Step 1 - Data Gathering: monitor executions of DM algorithms: system and algorithm-info
- Step 2 - Model Building: based on DM algorithm executions, input data, resources, faults etc.
- Step 3 - Online-Monitoring and Prediction: predict expected runtimes & progress and make suggestions for parameter settings
- Step 4 - Predictive Maintenance & feedback: adapt the execution and inform user of progress, unexpected behavior and suggestions as early as possible
- (Step 5 - Adaptation of the model) adapt parameters of the model to long term drift of the system behavior

C. *System Architecture*

In the following, an initial architecture is presented that enables data miners to build a system model for execution time prediction and performance fault-detection for distributed data mining algorithms. Fig. 4 depicts the architecture, which is currently being implemented [14].

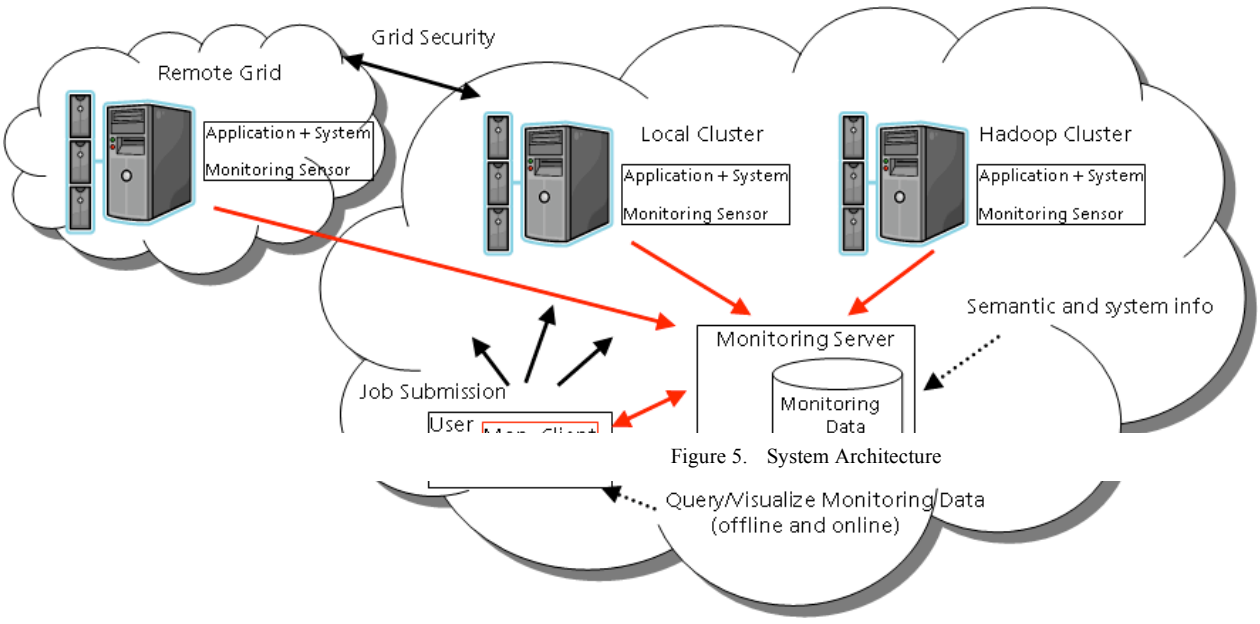

Figure 5. System Architecture

Figure 4. A semantic annotation file describing k-means

```
<Computation name="K-means" order="1">
  <Loop name="Runs Loop" level="1" type="unbounded" order="1">
    <ProgressMarker name="Runs Loop Iterator" level="2" type="multi_1to1"/>
    <Loop name="Optimizations Loop" level="2" type="unbounded" order="1">
      <ProgressMarker name="Opt. Loop Iterator" level="3" type="multi_1to1"/>
      <ProgressMarker name="Curr. Dist. Sum Value" level="3" type="multi_1to1"/>
    </Loop>
    <ProgressMarker name="Stable Distance Sum Value" level="2" type="multi_1to1"/>
  </Loop>
  <ProgressMarker name="Best Distance Sum Value" level="1" type="single"/>
</Computation>
```

Figure 5. A semantic annotation file describing k-means

Computing clusters hold the sensors that generate the monitoring data. We distinguish between system related monitoring data, which can be taken from standard tools, the operating system or middleware running on the machines, and application related monitoring data which contains semantic information on the execution of algorithms. A central monitoring server collects the monitoring data and merges it into a database.

The database holds the semantic and system information on which an analysis can take place offline and online. In the offline case, the monitoring data is analyzed in order to build the above-mentioned model of the system, if possible. Such a model could also include parameters, which are available only during the execution of an algorithm, for instance, that the execution of a loop can be estimated given that the first iterations have been performed and their execution time is known. Later, the generated model can be used online for presenting job status information including predictions to the user.

D. Semantic annotation of data mining algorithms

A key element of the approach is the semantic aggregation of collected monitoring data from semantically annotated data mining algorithms. This allows data collected from execution runs to be interpreted in the context of the monitored application. Semantic annotations are inserted as so-called progress markers into the algorithm according to semantic model of the algorithm, which is described by the algorithm developer in an XML-file [9]. We illustrate the approach by the means of an example: K-Means is a clustering algorithm that aims at partitioning a set of observations into k clusters where each observation belongs to the cluster with the nearest mean. The RapidMiner [10] implementation of the well known K-Means algorithm serves as example for the semantic instrumentation of data mining algorithms. The algorithm is structured as follows: an outer loop (runs) iterates over an inner process of cluster assignment. The inner process consists of a random initialization of cluster centroids and an inner loop (optimization) for a converging optimization of the cluster assignments. The progress markers in this scenario consist of the number of iterations of the different loops and a quality measure (distance value) of the cluster assignment. Fig. 5 shows the semantic description in an XML format. The K-Means algorithm can be distributed (see e.g. [1]) as the iterations of the outer loop represent different independent runs which can be computed in parallel.

V. CONCLUSIONS AND OUTLOOK

In this paper, we opened a novel research perspective on reliable distributed systems based on an approach of modeling distributed systems with data mining techniques. We introduced an initial architecture for observing the distributed systems and algorithm executions that allows for model building and on-line monitoring based on predicting upcoming reliability and performance problems with previously generated models. Users are enabled to take preventive measures for increased reliability or performance. We presented a concrete scenario of applying this approach in the field of distributed data mining algorithms.

Building an execution time and progress prediction model for some selected data mining algorithms being implemented in a specific distributed environment remains future work.

ACKNOWLEDGMENT

The authors gratefully acknowledge the financial support of the European Commission for the Project ACGT, FP6/2004/IST-026996.

REFERENCES

[1] Adranale, D.: Parallelization of the Weka Data Mining Toolkit on Hadoop. Master Thesis at Department of Distributed Systems, Faculty for Informatics, Otto-von-Guericke University of Magdeburg (2008)

[2] Cieslak, D.A.; Thain, D.; Chawla, N.V., "Short Paper: Troubleshooting Distributed Systems via Data Mining," *High Performance Distributed Computing, 2006 15th IEEE International Symposium on* , vol., no., pp. 309-312, June 19-23 2006

[3] Duan, R.; Prodan, R.; Fahringer, T., "Short Paper: Data Mining-based Fault Prediction and Detection on the Grid," *High Performance Distributed Computing, 2006 15th IEEE International Symposium on* , vol., no., pp.305-308

[4] Hofer, J. and Fahringer, T. 2007. Grid Application Fault Diagnosis Using Wrapper Services and Machine Learning. In *Proceedings of the 5th international Conference on Service-Oriented Computing* (Vienna, Austria, September 17 - 20, 2007). B. J. Krämer, K. Lin, and P. Narasimhan, Eds. Lecture Notes In Computer Science, vol. 4749. Springer-Verlag, Berlin, Heidelberg, 233-244

[5] Hoffmann, G.; Malek, M., "Call Availability Prediction in a Telecommunication System: A Data Driven Empirical Approach," *Reliable Distributed Systems, 2006. SRDS '06. 25th IEEE Symposium on* , vol., no., pp.83-95, 2-4 Oct. 2006

[6] Hoffmann, G.H., Trivedi, K.S., Malek, M.: A Best Practice Guide to Resource Forecasting for Computing Systems, IEEE Transactions on Reliability, vol. 56, No 4, Dec. 2004.

[7] Jing Tian; Zhi Yang; Wei Chen; Zhao, B.Y.; Yafei Dai, "Probabilistic Failure Detection for Efficient Distributed Storage Maintenance," *Reliable Distributed Systems, 2008. SRDS '08. IEEE Symposium on* , vol., no., pp.147-156, 6-8 Oct. 2008

[8] Kavulya, S.; Gandhi, R.; Narasimhan, P., "Gumshoe: Diagnosing Performance Problems in Replicated File-Systems," *Reliable Distributed Systems, 2008. SRDS '08. IEEE Symposium on* , vol., no., pp.137-146, 6-8 Oct. 2008

[9] Khayat, N.: Semantic Instrumentation and Measurement of Data Mining Algorithms, Technical Report on R&D 2, Hochschule Bonn-Rhein-Sieg, 2009.

[10] Mierswa, Ingo and Wurst, Michael and Klinkenberg, Ralf and Scholz, Martin and Euler, Timm: YALE: Rapid Prototyping for Complex Data Mining Tasks, in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), 2006.

[11] Mladenov, M.; Mock, M.; Grosspietsch, K.-E.: "Fault monitoring and correction in a walking robot using LMS filters," *Intelligent Solutions in Embedded Systems, 2008 International Workshop on* , vol., no., pp.1-10, 10-11 July 2008

[12] Nadeem, F., Prodan, R., and Fahringer, T. 2008. Characterizing, Modeling and Predicting Dynamic Resource Availability in a Large Scale Multi-purpose Grid. In *Proceedings of the 2008 Eighth IEEE international Symposium on Cluster Computing and the Grid* (May 19 - 22, 2008). CCGRID. IEEE Computer Society, Washington, DC, 348-357.

[13] Wegener, Dennis and Mock, Michael: Evaluating fault-tolerant data organization for data-intensive cluster applications. In Proc. of the IEEE SRDS 2008 Int. Workshop on Dependable Network Computing and Mobile Systems (DNCMS08), Naples, Italy, October, 2008.

[14] Wirth, P,: Monitoring von Data Mining Algorithm in verteilten Umgebungen, Master Thesis Hochschule Bonn-Rhein-Sieg (to be submitted).

[15] Zhang, R., Moyle, S., McKeever, S., and Bivens, A. 2007. Performance problem localization in self-healing, service-oriented systems using Bayesian networks. In *Proceedings of the 2007 ACM Symposium on Applied Computing* (Seoul, Korea, March 11 - 15, 2007). SAC '07. ACM, New York, NY, 104-109.