Data Mining Meets Network Management: The $\mathcal{NEMESIS}$ Project

Minos Garofalakis
Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974
minos@research.bell-labs.com

Rajeev Rastogi
Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974
rastogi@research.bell-labs.com

ABSTRACT

Modern communication networks generate large amounts of operational data, including traffic and utilization statistics and alarm/fault data at various levels of detail. These massive collections of *network-management* data can grow in the order of several Terabytes per year, and typically hide "knowledge" that is crucial to some of the key tasks involved in effectively managing a communication network (e.g., capacity planning and traffic engineering). In this short paper, we provide an overview of some of our recent and ongoing work in the context of the $\mathcal{NEMESIS}$ project at Bell Laboratories that aims to develop novel data warehousing and mining technology for the effective storage, exploration, and analysis of massive network-management data sets. We first give some highlights of our work on *Model-Based Semantic Compression (MBSC)*, a novel data-compression framework that takes advantage of attribute semantics and data-mining models to perform lossy compression of massive network-data tables. We discuss the architecture and some of the key algorithms underlying $\mathcal{SPARTAN}$, a model-based semantic compression system that exploits predictive data correlations and prescribed error tolerances for individual attributes to construct concise and accurate *Classification and Regression Tree (CaRT)* models for entire columns of a table. We also summarize some of our ongoing work on warehousing and analyzing network-fault data and discuss our vision of how data-mining techniques can be employed to help automate and improve fault-management in modern communication networks. More specifically, we describe the two key components of modern fault-management architectures, namely the *event-correlation* and the *root-cause analysis* engines, and propose the use of mining ideas for the automated inference and maintenance of the models that lie at the core of these components based on warehoused network data.

1. INTRODUCTION

Besides providing easy access to people and data around the globe, modern communication networks also *generate* massive amounts of operational data throughout their lifespan. As an example, Internet Service Providers (ISPs) continuously collect traffic and utilization information over their network to enable key network-management applications. This information is typically collected through monitoring tools that gather switch- and router-level data, such as SNMP/RMON probes [21] and Cisco's NetFlow measurement tools [1]. Such tools typically collect traffic data for each network element at fine granularities (e.g., at the level of individual packets or packet flows between source-destination pairs), giving rise to massive volumes of network-management data over time [6, 10]. Packet traces collected for traffic management in the Sprint IP backbone amount to 600 Gigabytes of data per day [10]! As another example, telecommunication providers typically generate and store records of information, termed "Call-Detail Records" (CDRs), for every phone call carried over their network. A typical CDR is a fixed-length record structure comprising several hundred bytes of data that capture information on various (categorical and numerical) attributes of each call; this includes network-level information (e.g., endpoint exchanges), time-stamp information (e.g., call start and end times), and billing information (e.g., applied tariffs), among others [5]. These CDRs are stored in tables that can grow to truly massive sizes, in the order of several Terabytes per year.

A key observation is that these massive collections of network-traffic and CDR data typically hide invaluable "knowledge" that enables several key network-management tasks, including application and user profiling, proactive and reactive resource management, traffic engineering, and capacity planning. Nevertheless, techniques for effectively managing these massive data sets and uncovering the knowledge that is so crucial to managing the underlying network are still in their infancy. Contemporary network-management tools do little more than elaborate report generation for all the data collected from the network, leaving most of the task of inferring useful knowledge and/or patterns to the human network administrator(s). As a result, effective network management is still viewed as more of an "art" known only to a few highly skilled (and highly sought-after) individuals. It is our thesis that, in the years to come, network management will provide an important application domain for very innovative, challenging and, at the same time, practically-relevant research in data mining and data warehousing.

This short abstract aims to provide an overview of our recent and ongoing research efforts in the context of $\mathcal{NEMESIS}$ (NEtwork-Management data warEhousing and analySIS) , a Bell Labs' research project that targets the development of novel data warehousing and mining technology for the effective storage, exploration, and analysis of massive network-management data sets. Our research agenda for $\mathcal{NEMESIS}$ encompasses several challenging research themes, including data reduction and approximate query processing [2, 7, 8], XML [12], effective network monitoring [4], mining techniques for network-fault management, and managing and analyzing continuous data streams. In this paper, we first give some highlights of our recent work on *Model-Based Semantic Compression (MBSC)*, a novel data-compression framework that takes advantage of attribute semantics and data-mining models to perform lossy compression of massive network-data tables. We also describe the architecture and some of the key algorithms underlying $\mathcal{SPARTAN}$, a system built based on the MBSC paradigm, that exploits predictive data correlations and prescribed error tolerances for individual attributes to construct concise and accurate *Classification and Regression Tree (CaRT)* models for entire columns of a table [2]. We then turn to our ongoing work on warehousing and analyzing network-fault data and discuss our vision of how data-mining techniques can be employed to help automate and improve fault-management in modern communication networks. More specifically, we describe the two key components of modern fault-management architectures, namely the *event-correlation* and the *root-cause analysis* engines, and offer some (more speculative) proposals on how mining ideas can be exploited for the automated inference and maintenance of the models that lie at the core of these components based on warehoused network data.

2. MODEL-BASED SEMANTIC COMPRESSION FOR NETWORK-DATA TABLES

Data compression issues arise naturally in applications dealing with massive data sets, and effective solutions are crucial for optimizing the usage of critical system resources like storage space and I/O bandwidth, as well as network bandwidth (for transferring the data) [5, 10]. Several statistical and dictionary-based compression methods have been proposed for text corpora and multimedia data, some of which (e.g., Lempel-Ziv or Huffman) yield provably optimal asymptotic performance in terms of certain ergodic properties of the data source. These methods, however, fail to provide adequate solutions for compressing massive data tables, such as the ones that house the operational data collected from large ISP and telecom networks. The reason is that all these methods view a table as a large byte string and do not account for the complex dependency patterns in the table. Compared to conventional compression problems, effectively compressing massive tables presents a host of novel challenges due to several distinct characteristics of table data sets and their analysis.

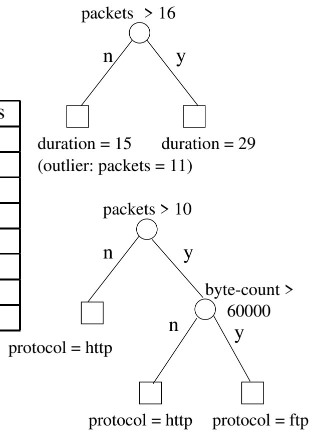
• **Semantic Compression.** Existing compression techniques are "syntactic" in the sense that they operate at the level of consecutive bytes of data. Such syntactic methods typically fail to provide adequate solutions for table-data compression, since they essentially view the data as a large byte string and do not exploit the complex dependency patterns in the table. Effective table compression mandates techniques that are *semantic* in nature, in the sense that they account for and exploit both (1) existing data dependencies and correlations among attributes in the table; and, (2) the meanings and dynamic ranges of individual attributes (e.g., by taking advantage of the specified error tolerances).

• **Approximate (Lossy) Compression.** Due to the exploratory nature of many data-analysis applications, there are several scenarios in which an exact answer may not be required, and analysts may in fact prefer a fast, approximate answer, as long as the system can guarantee an *upper bound on the error of the approximation*. For example, during a drill-down query sequence in ad-hoc data mining, initial queries in the sequence frequently have the sole purpose of determining the truly interesting queries and regions of the data table. Thus, in contrast to traditional lossless data compression, the compression of massive tables can often afford to be *lossy*, as long as some (user- or application-defined) upper bounds on the compression error are guaranteed by the compression algorithm. This is obviously a crucial differentiation, as even small error tolerances can help us achieve much better compression ratios.

In our recent work [2], we have proposed *Model-Based Semantic Compression (MBSC)*, a novel data-compression framework that takes advantage of attribute semantics and data-mining models to perform guaranteed-error, lossy compression of massive data tables. Abstractly, MBSC is based on the novel idea of exploiting data correlations and user-specified "loss"/error tolerances for individual attributes to construct concise data mining models and derive the best possible compression scheme for the data based on the constructed models. To make our discussion more concrete, we focus on the architecture and some of the key algorithms underlying $\mathcal{SPARTAN}$[1], a system that takes advantage of attribute correlations and error tolerances to build concise and accurate *Classification and Regression Tree (CaRT)* models [3] for entire columns of a table. More precisely, $\mathcal{SPARTAN}$ selects a certain subset of attributes (referred to as *predicted* attributes) for which no values are explicitly stored in the compressed table; instead, concise CaRTs that predict these values (within the prescribed error bounds) are maintained. Thus, for a predicted attribute X that is strongly correlated with other attributes in the table, $\mathcal{SPARTAN}$ is typically able to obtain a very succinct CaRT predictor for the values of X, which can then be used to completely eliminate the column for X in the compressed table. Clearly, storing a compact CaRT model in lieu of millions or billions of actual attribute values can result in substantial savings in storage.

protocol	duration	byte-count	packets
http	12	2,000	1
http	16	24,000	5
ftp	27	100,000	24
http	15	20,000	8
ftp	32	300,000	35
http	19	40,000	11
http	26	58,000	18
ftp	18	80,000	15

(a) Tuples in Table



(b) CaRT Models

Figure 1: Model-Based Semantic Compression.

EXAMPLE 2.1. *Consider the table with 4 attributes and 8 tuples shown in Figure 1(a), where each tuple represents a* data flow *in an IP network. The categorical attribute* protocol *records the application-level protocol generating the flow; the numeric attribute* duration *is the time duration of the flow; and, the numeric attributes* byte-count *and* packets *capture the total number of bytes and packets transferred. Let the acceptable errors due to compression for the numeric attributes* duration, byte-count, *and* packets *be 3, 1,000, and 1, respectively. Also, assume that the* protocol *attribute has to be compressed without error (i.e., zero tolerance). Figure 1(b) depicts a regression tree for predicting the* duration *attribute (with* packets *as the predictor attribute) and a classification tree for predicting the* protocol *attribute (with* packets *and* byte-count *as the predictor attributes). Observe that in the regression tree, the predicted value of duration (label value at each leaf) is almost always within 3, the specified error tolerance, of the actual tuple value. For instance, the predicted value of* duration *for the tuple with* packets = *1 is 15 while the original value is 12. The only tuple for which the predicted value violates this error bound is the tuple with* packets = *11, which is an marked as an outlier value in the regression tree. There are no outliers in the classification tree. By explicitly storing, in the compressed version of the table, each outlier value along with the CaRT models and the projection of the table onto only the predictor attributes (*packets *and* byte-count*), we can ensure that the error due to compression does not exceed the user-specified bounds. Further, storing the CaRT models (plus outliers) for* duration *and* protocol *instead of the attribute values themselves results in a reduction from 8 to 4 values for* duration *(2 labels for leaves + 1 split value at internal node + 1 outlier) and a reduction from 8 to 5 values for* protocol *(3 labels for leaves + 2 split values at internal nodes).* ■

To build an effective CaRT-based compression plan for the input data table, $\mathcal{SPARTAN}$ employs a number of sophisticated techniques from the areas of knowledge discovery and combinatorial optimization. Below, we list some of $\mathcal{SPARTAN}$'s salient features.

• **Use of Bayesian Network to Uncover Data Dependencies.** A Bayesian network is a directed acyclic graph (DAG) whose edges reflect strong predictive correlations between nodes of the graph [20]. $\mathcal{SPARTAN}$ uses a Bayesian network on the table's attributes to dramatically reduce

[1] [From Webster] **Spartan:** */'spart-*n/* (1) of or relating to Sparta in ancient Greece, (2) a: marked by strict self-discipline and avoidance of comfort and luxury, b: sparing of words : TERSE : LACONIC.

the search space of potential CaRT models since, for any attribute, the most promising CaRT predictors are the ones that involve attributes in its "neighborhood" in the network.

• **Novel CaRT-selection Algorithms that Minimize Storage Cost.** $\mathcal{SPARTAN}$ exploits the inferred Bayesian network structure by using it to intelligently guide the selection of CaRT models that minimize the overall storage requirement, based on the prediction and materialization costs for each attribute. We demonstrate that this model-selection problem is a strict generalization of the *Weighted Maximum Independent Set (WMIS)* problem [11], which is known to be $\mathcal{NP}$-hard. However, by employing a novel algorithm that effectively exploits the discovered Bayesian structure in conjunction with efficient, near-optimal WMIS heuristics, $\mathcal{SPARTAN}$ is able to obtain a good set of CaRT models for compressing the table.

• **Improved CaRT Construction Algorithms that Exploit Error Tolerances.** Since CaRT construction is computationally-intensive, $\mathcal{SPARTAN}$ employs the following three optimizations to reduce CaRT-building times: (1) CaRTs are built using random samples instead of the entire data set; (2) leaves are not expanded if values of tuples in them can be predicted with acceptable accuracy; (3) pruning is integrated into the tree growing phase using novel algorithms that exploit the prescribed error tolerance for the predicted attribute. $\mathcal{SPARTAN}$ then uses the CaRTs built to compress the full data set (within the specified error bounds) *in one pass*.

An extensive experimental study of the $\mathcal{SPARTAN}$ system with several real-life data tables has verified the effectiveness of our approach compared to existing syntactic (`gzip`) and semantic (fascicle-based [14]) compression techniques [2].

2.1 Overview of Approach

2.1.1 *Definitions and Notation*

The input to the $\mathcal{SPARTAN}$ system consists of a n-attribute table T, and a (user- or application-specified) n-dimensional vector of *error tolerances* $\bar{e} = [e_1, \ldots, e_n]$ that defines the *per-attribute* acceptable degree of information loss when compressing T. Let $\mathcal{X} = \{X_1, \ldots, X_n\}$ denote the set of n attributes of T and $dom(X_i)$ represent the domain of attribute X_i. Intuitively, e_i, the i^{th} entry of the tolerance vector $\bar{e}$ specifies an upper bound on the error by which any (approximate) value of X_i in the compressed table T_c can differ from its original value in T. For a numeric attribute X_i, the tolerance e_i defines an upper bound on the *absolute difference* between the actual value of X_i in T and the corresponding (approximate) value in T_c. That is, if x, x' denote the accurate and approximate value (respectively) of attribute X_i for *any* tuple of T, then our compressor guarantees that $x \in [x' - e_i, x' + e_i]$. For a categorical attribute X_i, the tolerance e_i defines an upper bound on the *probability* that the (approximate) value of X_i in T_c is different from the actual value in T. More formally, if x, x' denote the accurate and approximate value (respectively) of attribute X_i for *any* tuple of T, then our compressor guarantees that $P[x = x'] \geq 1 - e_i$. (Note that our error-tolerance semantics can also easily capture *lossless* compression as a special case, by setting $e_i = 0$ for all i.)

2.1.2 *Model-Based Semantic Compression*

Briefly, our proposed *model-based* methodology for semantic compression of data tables involves two steps: (1) exploiting data correlations and (user- or application-specified) error bounds on individual attributes to construct data mining models; and (2) deriving a good compression scheme using the constructed models. We define the model-based, compressed version of the input table T as a pair $T_c =< T', \{\mathcal{M}_1, \ldots, \mathcal{M}_p\} >$ such that T can be obtained from T_c within the specified error tolerance $\bar{e}$. Here, (1) T' is a small (possibly empty) projection of the data values in T that are retained *accurately* in T_c; and, (2) $\{\mathcal{M}_1, \ldots, \mathcal{M}_p\}$ is a set of data-mining models. A definition of our general model-based semantic compression problem can now be stated as follows.

[Model-Based Semantic Compression (MBSC)]: Given a multi-attribute table T and a vector of (per-attribute) error tolerances $\bar{e}$, find a set of models $\{\mathcal{M}_1, \ldots, \mathcal{M}_m\}$ and a compression scheme for T based on these models $T_c =< T', \{\mathcal{M}_1, \ldots, \mathcal{M}_p\} >$ such that the specified error bounds $\bar{e}$ are not exceeded and the storage requirements $|T_c|$ of the compressed table are minimized. ■

Given the multitude of possible models that can be extracted from the data, the general MBSC problem definition covers a huge design space of possible alternatives for semantic compression. We now provide a more concrete statement of the problem addressed in our work on the $\mathcal{SPARTAN}$ system.

[$\mathcal{SPARTAN}$ CaRT-Based Semantic Compression]: Given a multi-attribute table T with a set of attributes $\mathcal{X}$, and a vector of (per-attribute) error tolerances $\bar{e}$, find a subset $\{X_1, \ldots, X_p\}$ of $\mathcal{X}$ and a set of corresponding CaRT models $\{\mathcal{M}_1, \ldots, \mathcal{M}_p\}$ such that: (1) model $\mathcal{M}_i$ is a predictor for the values of attribute X_i based solely on attributes in $\mathcal{X} - \{X_1, \ldots, X_p\}$, for each $i = 1, \ldots, p$; (2) the specified error bounds $\bar{e}$ are not exceeded; and, (3) the storage requirements $|T_c|$ of the compressed table $T_c =< T', \{\mathcal{M}_1, \ldots, \mathcal{M}_p\} >$ are minimized. ■

Abstractly, $\mathcal{SPARTAN}$ seeks to partition the set of input attributes $\mathcal{X}$ into a set of *predicted attributes* $\{X_1, \ldots, X_p\}$ and a set of *predictor attributes* $\mathcal{X} - \{X_1, \ldots, X_p\}$ such that the values of each predicted attribute can be obtained within the specified error bounds based on (a subset of) the predictor attributes through a small classification or regression tree (except perhaps for a small set of outlier values). Note that we do not allow a predicted attribute X_i to also be a predictor for a different attribute. This restriction is important since predicted values of X_i can contain errors, and these errors can cascade further if the erroneous predicted values are used as predictors, ultimately causing error constraints to be violated. The final goal, of course, is to minimize the overall storage cost of the compressed table. This storage cost $|T_c|$ is the sum of two basic components:

1. *Materialization cost*, i.e., the cost of storing the values for all predictor attributes $\mathcal{X} - \{X_1, \ldots, X_p\}$. This cost is represented in the T' component of the compressed table, which is basically the projection of T onto the set of predictor attributes. The storage cost of materializing attribute X_i is denoted by $\texttt{MaterCost}(X_i)$.

2. *Prediction cost*, i.e., the cost of storing the CaRT models used for prediction plus (possibly) a small set of outlier values of the predicted attribute for each model. (We use the notation $\mathcal{X}_i \rightarrow X_i$ to denote a CaRT predictor for attribute X_i using the set of predictor attributes $\mathcal{X}_i \subseteq \mathcal{X} - \{X_1, \ldots, X_p\}$.) The storage cost of predicting attribute X_i using the CaRT predictor $\mathcal{X}_i \rightarrow X_i$ is denoted by $\texttt{PredCost}(\mathcal{X}_i \rightarrow X_i)$; this does *not* include the cost of materializing the predictor attributes in $\mathcal{X}_i$.

2.2 $\mathcal{SPARTAN}$ System Architecture

As depicted in Figure 2, the architecture of the $\mathcal{SPARTAN}$ system comprises of four major components: the DEPENDENCYFINDER, the CARTSELECTOR, the CARTBUILDER, and the ROWAGGREGATOR. In the following, we provide a brief overview of each $\mathcal{SPARTAN}$ component; for a more detailed description of each component and the relevant algorithms, the interested reader is referred to [2].

• DEPENDENCYFINDER. The purpose of the DEPENDENCYFINDER component is to produce an *interaction model* for the input table at-

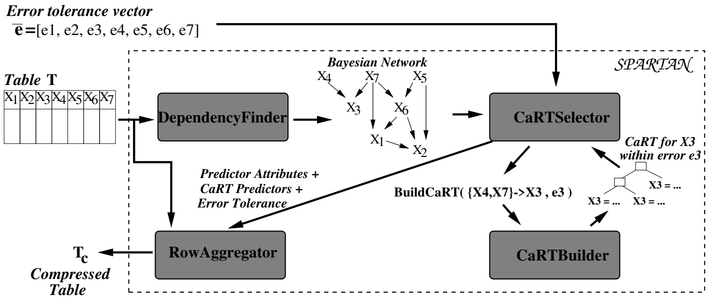


Figure 2: $\mathcal{SPARTAN}$ System Architecure.

tributes, that is then used to guide the CaRT building algorithms of $\mathcal{SPARTAN}$. The main observation here is that, since there is an exponential number of possibilities for building CaRT-based attribute predictors, we need a concise model that identifies the strongest correlations and "predictive" relationships in the input data.

The approach used in the DEPENDENCYFINDER component of $\mathcal{SPARTAN}$ is to construct a *Bayesian network* [20] on the underlying set of attributes $\mathcal{X}$. Abstractly, a Bayesian network imposes a Directed Acyclic Graph (DAG) structure G on the set of nodes $\mathcal{X}$, such that directed edges capture direct statistical dependence between attributes. (The exact dependence semantics of G are defined shortly.) Thus, intuitively, a set of nodes in the "neighborhood" of X_i in G (e.g., X_i's parents) captures the attributes that are strongly correlated to X_i and, therefore, show promise as possible predictor attributes for X_i.

• <u>CARTSELECTOR.</u> The CARTSELECTOR component constitutes the core of $\mathcal{SPARTAN}$'s model-based semantic compression engine. Given the input table T and error tolerances e_i, as well as the Bayesian network on the attributes of T built by the DEPENDENCYFINDER, the CARTSELECTOR is responsible for selecting a collection of predicted attributes and the corresponding CaRT-based predictors such that the final overall storage cost is minimized (within the given error bounds). As discussed above, $\mathcal{SPARTAN}$'s CARTSELECTOR employs the Bayesian network G built on $\mathcal{X}$ to intelligently guide the search through the huge space of possible attribute prediction strategies. Clearly, this search involves repeated interactions with the CARTBUILDER component, which is responsible for actually building the CaRT-models for the predictors (Figure 2).

We demonstrate that even in the simple case where the set of nodes that is used to predict an attribute node in G is *fixed*, the problem of selecting a set of predictors that minimizes the combination of materialization and prediction cost naturally maps to the *Weighted Maximum Independent Set (WMIS)* problem, which is known to be $\mathcal{NP}$-hard and notoriously difficult to approximate [11]. Based on this observation, we propose a CaRT-model selection strategy that starts out with an initial solution obtained from a near-optimal heuristic for WMIS [13] and tries to incrementally improve it by small perturbations based on the unique characteristics of our problem. We also give an alternative *greedy* model-selection algorithm that chooses its set of predictors using a simple local condition during a single "roots-to-leaves" traversal of the Bayesian network G.

• <u>CARTBUILDER.</u> Given a collection of predicted and (corresponding) predictor attributes $\mathcal{X}_i \rightarrow X_i$, the goal of the CARTBUILDER component is to efficiently construct CaRT-based models for each X_i in terms of $\mathcal{X}_i$ for the purposes of semantic compression. Induction of CaRT-based models is typically a computation-intensive process that requires multiple passes over the input data [3, 17]. As we demonstrate, however, $\mathcal{SPARTAN}$'s CaRT construction algorithms can take advantage of the compression semantics and exploit the user-defined error-tolerances to effectively prune computation. In addition, by building CaRTs using data samples instead of the entire data set, $\mathcal{SPARTAN}$ is able to further speed up model construction.

• <u>ROWAGGREGATOR.</u> Once $\mathcal{SPARTAN}$'s CARTSELECTOR component has finalized a "good" solution to the CaRT-based semantic compression problem, it hands off its solution to the ROWAGGREGATOR component which tries to further improve the compression ratio through row-wise clustering. Briefly, the ROWAGGREGATOR uses a fascicle-based algorithm [14] to compress the predictor attributes, while ensuring (based on the CaRT models built) that errors in the predictor attribute values are not propagated through the CaRTs in a way that causes error tolerances (for predicted attributes) to be exceeded.

2.3 Future Directions in MBSC

We strongly believe that our research on the $\mathcal{SPARTAN}$ system has only scratched the surface of model-based semantic compression and its potential applications in managing and exploring network management data. We are currently extending our work on MBSC in a number of different directions.

• **MBSC for Continuous Data Streams: The Online-$\mathcal{SPARTAN}$ System.** Network-management systems collect and store *continuous streams of data*. For example, ISPs continuously collect packet- and flow-level traces of traffic in their network to enable sophisticated network traffic management applications such as traffic demand computation, capacity planning and provisioning, and determining pricing plans [6]. This data arrives at the network-management station as a continuous stream and archived in massive and fast-growing data tables. The goal of the Online-$\mathcal{SPARTAN}$ system is to build on the MBSC ideas of $\mathcal{SPARTAN}$ to effectively compress such continuous data streams. Applying MBSC over streaming data introduces a number of novel technical challenges.

More specifically, the data characteristics (e.g., attribute correlations) of streams can vary over time. $\mathcal{SPARTAN}$'s methodology of capturing the characteristics of the input table using data mining models and deriving a good compression plan from the constructed models would have to become *online* and *adaptive* for the data-stream scenario. This means that the models have to be maintained online as new tuples arrive over the data stream [9] and the compression plan should adapt appropriately to changes in data characteristics. The goal, of course, is to maintain the best possible compressed representation of the data seen thus far.

• **Distributed MBSC Plans.** Many traffic management applications (e.g., traffic demand computation, capacity planning [6]) for ISP networks need a global view of the network traffic, requiring data to be collected from multiple monitoring points distributed over the network [4]. Given the massive volume of collected data, it is highly desirable to compress that data before it is transferred to data warehouses and repositories for processing and archival [10]. Suppose that MBSC is the compression method of choice for the data collected at each monitoring point. One approach would be to build data mining models based solely on the locally-collected data. However, these models are not guaranteed to capture the characteristics of the overall network traffic and might, therefore, result in compression schemes that are suboptimal for the overall traffic data – more sophisticated approaches are needed. Note that this problem relates to problems studied in the context of *distributed data mining*, especially to the problem of combining/consolidating mining models built using portions of a distributed data collection [16].

• **New Techniques for MBSC.** The basic thesis of MBSC is that the model, or combination of models, which compresses a data table optimally, is dependent on the data table. The current implementation of $\mathcal{SPARTAN}$, which uses CaRT-based column-wise compression, followed by fascicles-based row-wise compression, is a specific instantiation of the general MBSC philosophy. More sophisticated combinations of row-wise and column-wise models are possible. As an example, $\mathcal{SPARTAN}$'s fascicles-based ROWAGGREGATOR component can be replaced by a column-wise differential-coding-based component [18, 5]. In the long run, we envision $\mathcal{SPARTAN}$ as a *self-tuning* semantic compression system that derives the best compression scheme from the input data table, without being restricted to a specific family of models.

Currently $\mathcal{SPARTAN}$ exploits *data semantics* in two ways: (1) by capturing data correlations; and, (2) by using user- or application-specified per-attribute error tolerances which specify the acceptable degree of information loss. Developing other interfaces to specify data semantics in a general fashion, and also MBSC techniques for exploiting these interfaces is an interesting open problem. It is desirable to keep the interface general and independent of specific nature of the data so that the underlying MBSC tools and techniques are widely applicable.

3. DATA MINING TECHNIQUES FOR NETWORK-FAULT MANAGEMENT

Modern communication networks have evolved into highly complex systems, typically comprising large numbers of interconnected elements (e.g., routers, switches, bridges) that work together to provide end-users with various data and/or voice services. This increase in system scale and the number of elements obviously implies an increased probability of faults occurring somewhere in the network. Further, the complex inter-dependencies that exist among the various elements in the network cooperating to provide some service imply that a fault can propagate widely, causing *floods* of alarm signals from very different parts of the network. As an example, a switch failure in an IP network can cause the network to be partitioned resulting in alarms emanating from multiple elements in different network partitions and subnets, as they detect that some of their peers are no longer reachable. To deal with these situations, modern network-management platforms provide certain *fault-management* utilities that try to help administrators make sense of alarm floods, and allow them to quickly and effectively zero in on the root cause of the problem.

Typically, the architecture of a fault-management subsystem comprises two key components: the *Event Correlator (EC)* and the *Root-Cause Analyzer (RCA)*, as depicted in Figure 3. The goal of the Event Correlator is improve the information content of the observed events by filtering out uninteresting, "secondary" alarms from the alarm flood arriving at the network-management station [15, 22]. (Secondary alarms or *symptoms* are observable events that are directly caused by other events observed in the network.) This filtering is implemented with the help of a set of *fault-propagation rules* that the Event Correlator uses to model the propagation of alarm signals in the underlying network. The output of the Event Correlator, i.e., the "primary" alarm signals in the observed set of alarms, are then fed into the Root-Cause Analyzer whose goal is to produce a set of possible root causes for the observed problem along with associated degrees of confidence for each "guess" (Figure 3).

The fault-propagation rules that model the propagation of alarms throughout the underlying network form the basic core of the Event Correlation engine. In general, these rules try to capture the *probabilistic causal relationships* that exist between the various alarm signals in the network. As an example, Figure 4 depicts a small subset of such fault-propagation rules; based on the figure, alarm signal A_1 causes the occurrence of alarm signal A_3 with probability p_{13} and that of alarm signal A_4 with probability p_{14}. Thus, the fault-propagation

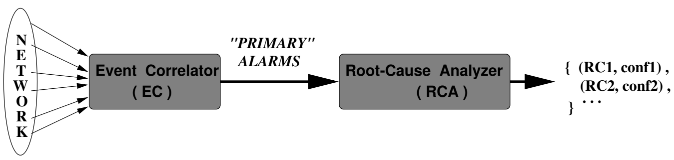


Figure 3: Fault-Management System Architecture.

rules that lie at the heart of the Event Correlator are essentially equivalent to a *causal Bayesian model* [19, 20] for network alarm signals. Given such a causal model for network alarms, the problem of filtering out secondary events in the Event Correlator can be formulated as an optimization problem in a variety of interesting ways. For example, a possible formulation would be as follows: Given a confidence threshold $\theta \in (0, 1)$ and the set of all observed alarm signals A, find a minimal subset P of A such that $P[A|P] > \theta$ (i.e., the probability that A was actually "caused" by P exceeds the desired confidence θ).

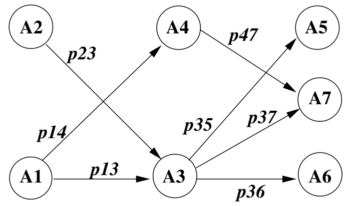


Figure 4: Example Fault-Propagation Model for EC.

Current State-of-the-Art. There are several commercially-available products that offer event-correlation services for data-communication networks. Examples include SMARTS InCharge [22], the Event-Correlation Services (ECS) component of the HP OpenView network-management platform, CISCO's InfoCenter, GTE's Impact, and so on. A common characteristic of all these Event Correlators is that they essentially force the network administrator(s) to "hand-code" the fault-propagation rules for the underlying network using either a language-based or a graphics-based specification tool. This is clearly a very tedious and error-prone process for any large-scale IP network comprising hundreds or thousands of heterogeneous, multi-vendor elements. Furthermore, it is non-incremental since a large part of the specification may need to be changed when the topology of the network changes or new network elements are introduced. We believe that such solutions are simply inadequate for tomorrow's large-scale, heterogeneous, and highly-dynamic IP networking environments.

Our Proposed Approach. Rather than relying on human operators to "hand-code" the core of the Event-Correlation engine, we propose the use of data-mining techniques to help automate the task of inferring and incrementally maintaining the causal model of network alarm signals (Figure 5). For the inference task (typically performed off-line), our data-mining tool can exploit the database of alarm signals collected and stored over the lifespan of the network along with important "domain-specific knowledge" (e.g., network topology and routing-protocol information) to automatically construct the correct causal model of fault-propagation rules. For the maintenance task (typically performed on-line), our data-mining tool can again exploit such "domain-specific knowledge" along with information on network updates (e.g., topology changes or new additions to the network) and the incoming stream of network alarm signals to automatically effect the appropriate updates to the fault-propagation model.

We should note here that, even though the problem of inferring causal Bayesian models from data has been studied for some time

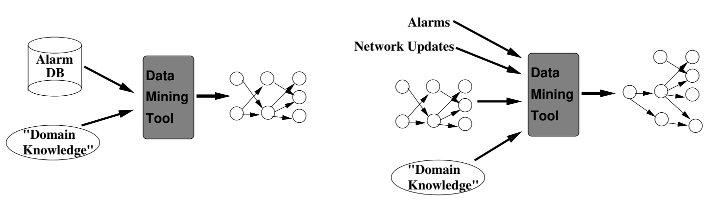


Figure 5: Exploiting Data Mining for Automated EC Inference and Maintenance.

in the data-mining and machine-learning communities [19, 20], the automatic extraction of event-correlation models for communication networks presents a host of new challenges due to several unique characteristics of the problem domain. First, the issue of how to effectively incorporate and exploit important "domain-specific knowledge" (like the network topology or routing-protocol information) in the model-learning algorithm is certainly very challenging and non-trivial. Second, it is important to incorporate the *temporal aspects* of network alarm signals in the data-mining process; for example, alarms that occur within a small time window are more likely to be correlated than alarms separated by larger amounts of time. Finally, the learning algorithm needs to be *robust* to lost or spurious alarm signals, both of which are common phenomena in modern communication networks.

For the Root-Cause Analyzer, data-mining techniques can again be exploited; for example, our tools can use failure data collected from the field to automatically learn failure "signatures" and map them to an associated root cause. Once again, it is crucial to effectively incorporate important "domain-specific knowledge" (like the network topology) in the data-mining process.

4. CONCLUSIONS

Operational data collected from modern communication networks is massive and hides "knowledge" that is invaluable to several key network-management tasks. In this short abstract, we have provided an overview of some of our recent and ongoing work in the context of the $\mathcal{NEMESIS}$ project at Bell Laboratories that aims to develop novel data warehousing and mining technology for the effective storage, exploration, and analysis of massive network-management data sets. We have given some key highlights of our recent work on *Model-Based Semantic Compression (MBSC)*, a novel data-compression framework that takes advantage of attribute semantics and data-mining models to perform lossy compression of massive network-data tables. We have also summarized our vision of how data-mining techniques can be employed to help automate and improve fault-management in modern communication networks. We firmly believe that, in the years to come, network management will provide an important application domain for very innovative, challenging and, at the same time, practically-relevant research in data mining and data warehousing.

Acknowledgements: Many thanks to Shivnath Babu (our coauthor in [2]) for his valuable contributions on the $\mathcal{SPARTAN}$ semantic-compression engine.

5. REFERENCES

[1] "NetFlow Services and Applications". Cisco Systems White Paper, 1999.

[2] Shivnath Babu, Minos Garofalakis, and Rajeev Rastogi. "SPARTAN: A Model-Based Semantic Compression System for Massive Data Tables". In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, Santa Barbara, California, May 2001.

[3] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *"Classification and Regression Trees"*. Chapman & Hall, 1984.

[4] Yuri Breitbart, Chee-Yong Chan, Minos Garofalakis, Rajeev Rastogi, and Avi Silberschatz. "Efficiently Monitoring Bandwidth and Latency in IP Networks". In *Proceedings of IEEE INFOCOM'2001*, Anchorage, Alaska, April 2001.

[5] Adam L. Buchsbaum, Donald F. Caldwell, Kenneth Church, Glenn S. Fowler, and S. Muthukrishnan. "Engineering the Compression of Massive Tables: An Experimental Approach". In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, California, January 2000.

[6] R. Caceres, N. Duffield, A. Feldmann, J. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. Kalmanek, B. Krishnamurthy, D. Lavelle, P. Mishra, K.K. Ramakrishnan, J. Rexford, F. True, and J. van der Merwe. "Measurement and analysis of IP network usage and behavior". *IEEE Communications Magazine*, May 2000.

[7] Kaushik Chakrabarti, Minos Garofalakis, Rajeev Rastogi, and Kyuseok Shim. "Approximate Query Processing Using Wavelets". In *Proceedings of the 26th International Conference on Very Large Data Bases*, pages 111–122, Cairo, Egypt, September 2000.

[8] Amol Deshpande, Minos Garofalakis, and Rajeev Rastogi. "Independence is Good: Dependency-Based Histogram Synopses for High-Dimensional Data". In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, Santa Barbara, California, May 2001.

[9] Pedro Domingos and Geoff Hulten. "Mining high-speed data streams". In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 71–80, Boston, Massachusetts, August 2000.

[10] C. Fraleigh, S. Moon, C. Diot, B. Lyles, and F. Tobagi. "Architecture of a Passive Monitoring System for Backbone IP Networks". Technical Report TR00-ATL-101-801, Sprint Advanced Technology Laboratories, October 2000.

[11] M.R. Garey and D.S. Johnson. *"Computers and Intractability: A Guide to the Theory of NP-Completeness"*. W.H. Freeman, 1979.

[12] Minos N. Garofalakis, Aristides Gionis, Rajeev Rastogi, S. Seshadri, and Kyuseok Shim. "XTRACT: A System for Extracting Document Type Descriptors from XML Documents". In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 165–176, Dallas, Texas, May 2000.

[13] Magnús M. Halldórsson. "Approximations of Weighted Independent Set and Hereditary Subset Problems". *Journal of Graph Algorithms and Applications*, 4(1):1–16, 2000.

[14] H.V. Jagadish, Jason Madar, and Raymond Ng. "Semantic Compression and Pattern Extraction with Fascicles". In *Proceedings of the 25th International Conference on Very Large Data Bases*, pages 186–197, Edinburgh, Scotland, September 1999.

[15] Gabriel Jakobson and Mark D. Weissman. "Alarm Correlation". *IEEE Network*, pages 52–59, November 1993.

[16] H. Kargupta and P. Chan. *"Advances in Distributed and Parallel Knowledge Discovery"*. MIT Press, Cambridge, Massachusetts, 2000.

[17] Yasuhiko Morimoto, Hiromu Ishii, and Shinichi Morishita. "Efficient Construction of Regression Trees with Range and Region Splitting". In *Proceedings of the 23rd International Conference on Very Large Data Bases*, Athens, Greece, August 1997.

[18] Wee Keong Ng and Chinya V. Ravishankar. "Block-Oriented Compression Techniques for Large Statistical Databases". *IEEE Transactions on Knowledge and Data Engineering*, 9(2):314–328, March 1997.

[19] Judea Pearl. *"Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference"*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1988.

[20] Judea Pearl. *"Causality – Models, Reasoning, and Inference"*. Cambridge University Press, 2000.

[21] William Stallings. *"SNMP, SNMPv2, SNMPv3, and RMON 1 and 2"*. Addison-Wesley Longman, Inc., 1999. (Third Edition).

[22] S. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. "High Speed & Robust Event Correlation". *IEEE Communications Magazine*, May 1996.