Condition Data Aggregation with Application to Failure Rate Calculation of Power Transformers

Jyotishman Pathak[†]   **Yong Jiang**[‡]   **Vasant Honavar**[†]   **James McCalley**[‡]

[†]Department of Computer Science, Iowa State University, Ames, IA 50011-1040, USA

[‡]Department of Electrical & Computer Engineering, Iowa State University, Ames, IA 50011-1040, USA

{jpathak, jiangy, honavar, jdm}@iastate.edu

Abstract

Cost-effective equipment maintenance for electric power transmission systems requires ongoing integration of information from multiple, highly distributed, and heterogeneous data sources storing various information about equipment. This paper describes a federated, query-centric data integration and knowledge acquisition framework for condition monitoring and failure rate prediction of power transformers. Specifically, the system uses substation equipment condition data collected from distributed data resources, some of which may be local to the substation, to develop Hidden Markov Models (HMMs) which transform the condition data into failure probabilities. These probabilities provide the most current knowledge of equipment deterioration, which can be used in system-level simulation and decision tools. The system is illustrated using dissolved gas-in-oil field data for assessing the deterioration level of power transformer insulating oil.

Keywords: Data Integration, Hidden Markov Models, Transformer Failure Mode Estimation.

1 Introduction

The advancements in electric power systems, power transmission and distribution grids are critical for a nation's growth and development. However, they are comprised of a large number of highly distributed and capital-intensive physical assets that can fail in catastrophic ways. The reliability and proper functioning of these assets are dependent on effective approaches for problems related to their operation and maintenance. Quality of these solutions depend not only on the quality of the information used for assessment, but also on how it is processed. Central, and essential, are information characterizing the health or condition of the assets. For example, equipment *age* and *time* since the last inspection and maintenance are widely used asset condition indicators. As a result, 'nameplate' data and maintenance histories are often used in the decision-making process. Until recently, the coordination of this information was human-driven, which is not only tedious and time-consuming, but also costly. However, due to the recent developments in sensing, communications, distributed computing and database technologies, it has become feasible for decision-makers to access operating histories and asset-specific real-time monitoring of data. Creative use of this data via processing, integration, assessment, and decision algorithms can significantly enhance the quality of the final actions taken, and result in very large national impact in terms of more economic and reliable system performance.

Against this background, in this paper we investigate a federated, query-centric approach to information integration and knowledge acquisition from autonomous, distributed, and heterogeneous data sources for condition monitoring and failure mode estimation of power transformers. These data sources may include intelligent electronic devices (IEDs) local to the equipment or data repositories in corporate servers. Unavoidably in real life situations, the related data sources maintained by different institutions often differ in structure, organization, query capabilities, and more importantly ontological commitments [17] - assumptions concerning the *objects* that exist in the *world*, the *properties* of the objects and their possible *values*, *relationships* between them, and their intended *meaning*. In other words, data sources often do not agree on using a shared vocabulary of terms and concepts in a coherent and consistent manner. As a result, it becomes increasingly difficult for individuals and autonomous software entities to seamlessly query the data sources or assert facts about them. Our approach to this problem has resulted in the adaptation of a system called

INDUS (Intelligent Data Understanding System) [26]. INDUS[1] imposes a clear separation between the data and semantics (or intended meaning) of data, which allows the users to reconcile semantic differences between multiple heterogeneous data sources from their own point of view. With the help of specific software wrappers, the system exposes autonomous data sources (regardless of their location, internal structure, and query interfaces) as though they were relational databases (i.e., a collection of inter-related tables), structured according to an ontology supplied by the user. INDUS when equipped with data mining and decision-making algorithms for ontology-driven knowledge acquisition can accelerate the pace of discovery in many data-rich domains. Specifically for this paper, we used INDUS to integrate power transformer condition data for training Hidden Markov Models [25], a model effective in characterizing discrete state random processes where the mapping between states (deterioration levels in this case) and observations is uncertain.

The rest of the paper is structured as follows: Section 2 describes the data integration component of INDUS, whereas a detailed description of failure rate probability estimation using HMM is given in Section 3. In Section 4 we describe the implementation details of our framework and show how transformer failure rate can be estimated from condition monitoring data. Finally, we summarize our work and provide a brief discussion about future work in Section 5.

2 Data Integration in INDUS

The estimation of the state of an asset (e.g., transformer, circuit breaker, underground cable, insulator, etc.), is typically made using a variety of data. In general, there may be up to four classes of this data: *equipment data, operating histories, maintenance histories,* and *condition histories.* The *equipment data* comprises the so-called 'nameplate' information including manufacturer, make, model, rated currents, voltages, and powers, equipment's age, and manufacturer's recommended maintenance schedule. The *operating histories* capture the electrical and environmental conditions to which the equipment has been subjected in the past, e.g., temperatures, loading histories and through faults for transformers, and operations and I^2t for circuit breakers. The *maintenance histories* contain records of all inspections and maintenance activities performed on each piece of equipment. *Condition histories* are comprised of measurements providing in-

formation about the state of the equipment with respect to one or more failure modes. Common condition data information for a transformer includes that coming from tests on: oil (dissolved gas, moisture, hydrogen, and furan), power factor, winding resistance, partial discharge (acoustic emissions, spectral decomposition of currents), and infrared emissions. All of this data can be collected either manually via inspections/laboratory testings or using continuous monitoring sensors. Usually, these four classes of information are maintained in multiple database systems distributed between the substation and corporate headquarters using various commercially available storage technologies (e.g., Oracle) together with a variety of data standards and proprietary systems. Effective use of this data demands for versatile data integration and management systems for efficiently extracting relevant information.

In practice, data integration systems [3,13,16,20,21, 23] attempt to provide users with seamless and flexible access to information from autonomous, distributed, and heterogeneous data sources through a unified query interface. Ideally, such systems should allow the users to specify *what* information is needed instead of *how* it can be obtained. In other words, it should provide mechanisms for:

- Specification of a query expressed in terms of a user-specified vocabulary (ontology).

- Specifying mappings between user ontology and data-source ontologies.

- Automatically transforming user queries into queries that can be answered/understood by the respective data sources.

- Hiding the complexity of communication and interaction with heterogeneous, distributed data sources.

- Mapping the results obtained into the form expected by the user and storing them for future analysis.

- Allowing effortless incorporation of new data sources as needed, and supporting sharing of ontologies between different users.

In general, there are two broad approaches to data integration: *Data Warehousing* and *Database Federation.* In the data warehousing approach, data from heterogeneous information sources is gathered, mapped to a common structure and stored in a central location. Periodic updates are required to ensure that the information contained in the warehouse is up-to-date with

---

[1]The acronym INDUS should not be confused with a suite of commercial service delivery and asset management solutions provided by Indus (www.indus.com).

the contents of the individual sources. However, the data replication/updating process can be quite expensive in case of large information repositories. Also, this approach relies on a single common ontology for all users which is specified as part of the warehouse design. As a result, the system tends to be less flexible. On the other hand, in case of database federation, the information needed to answer a query is gathered directly from the data sources in response to the posted query. Hence, the results are up-to-date with respect to the contents of the data sources at the time the query is posted. More importantly, this approach is being more readily adapted to applications where users are able to impose their own ontologies and specify queries using the various concepts in those ontologies. Because our focus is on data integration for scientific applications, which requires users to be able to flexibly interpret and integrate data from multiple autonomous sources, we adopt the federated architecture for our system.

Typically, a query posted by the user must be decomposed into a set of operations corresponding to the information that needs to be gathered from each data source and the form in which this information must be returned to the system. These operations should be capable of dealing with syntactic (or structural) and semantic (or intended meaning) mismatches by transforming the queries expressed in terms of the user ontology into data source-specific execution plans. These plans describe what information to extract from each data source and how to combine the results. In general, there are two basic approaches for dealing with semantic mismatches for query answering: *Source-Centric approach* and *Query-Centric approach*. In the case of the source-centric approach, each individual data source determines how the terms in a data source ontology (or vocabulary) are mapped to terms in the user (or global) ontology. Thus, the user has little control on the true meaning of concepts, and hence the results of a query. In other words, this approach puts the information sources in control of the semantics. In contrast, in the query-centric approach to information integration, concepts in the user ontology are defined in terms of concepts in data source-specific ontologies. Thus, the query-centric approach is better suited for data integration applications in which the users need the ability to impose the ontologies (and semantics) of their choice to flexibly interpret and analyze information from autonomous sources. However, this requires the user or administrator of the integration system to specify precisely how concepts in the user ontology are mapped to data source ontologies. As a result, the user controls/specifies the semantics because of of which we adopt the query-centric approach to data integration
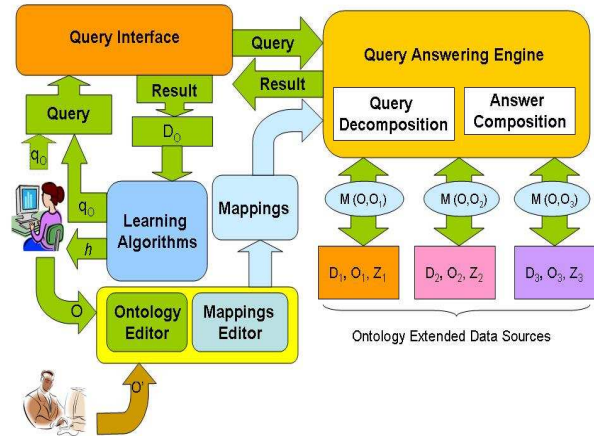
in INDUS.

Figure 1. Simplified INDUS Architecture

A simplified architecture of INDUS is shown in Figure 1. Typically, several related distributed and semantically heterogeneous *data sources* can be available to *users* who may want to query the data sources of interest via a *query interface*. Each user has the ability to impose his or her semantics by defining *user ontologies*. The system provides an user-friendly *ontology* and *mapping editor* [6] via which the users of the system can specify mappings between the concepts in the user ontology and data source ontologies. These ontologies and mappings are stored in the *mapping repository*. Once a query is posed by the user, it is handled by the *query answering engine* which acts as a middleware between the users (or clients) and data sources (or servers). This engine has access to the data sources as well as the set of user-specified mappings. Thus, when the engine receives an user query, it decomposes the query into distributed sources, maps the individual queries into data source-specific semantics, and finally composes the partial answers of each sub-query into final result which is sent back to the user.

There are several features that distinguish INDUS from various other data integration systems:

- INDUS imposes a clear separation between data and the semantics of data. Such an approach allows users to specify mappings from the concepts in their ontologies to the data source ontologies.

- Instead of having a single global ontology (common to all users), INDUS allows users to specify their ontologies and mappings to the data source ontologies.

3

- INDUS can be hooked up with various knowledge acquisition and decision-making algorithms (e.g., data mining algorithms) whose information requirements can be formulated as statistical[2] queries [9].

We discuss these features in the remainder of this section.

2.1 Ontology-Extended Data Sources

Assume that we have a set of physically distributed data sources, $D_1, \cdots, D_n$, such that each data source D_i contains only a fragment of the whole data D. In general, two common types of data fragmentation are defined [11]: *horizontal fragmentation*, where each data fragment contains a subset of data tuples, and *vertical fragmentation*, where each data fragment contains subtuples of data tuples. However, one can envision a combination of the two types of data fragmentation, and also more general relational data fragmentations.

Formally, an ontology can be defined as a specification of *objects, categories, properties* and *relationships* used to conceptualize some domain of interest [17]. Let D_i be a distributed data source described by the set of attributes $\{A_1^i, \cdots, A_m^i\}$ and $O_i = \{\Gamma_1^i, \cdots, \Gamma_m^i\}$ an ontology associated with the data source. The element $\Gamma_j^i \in O_i$ corresponds to the attribute A_j^i and defines the type of that particular attribute. These types can be either linear (e.g., String, Integer etc.), or an ordering (or *hierarchy* [9]) of a set of terms (e.g., attribute value taxonomies). The schema S_i of a data source D_i is given by the set of attributes $\{A_1^i, \cdots, A_m^i\}$ used to describe the data, together with their respective attribute types $\{\Gamma_1^i, \cdots, \Gamma_m^i\}$, defined by the ontology O_i, i.e., $S_i = \{A_1^i : \Gamma_1^i, \cdots, A_n^i : \Gamma_n^i\}$.

We define an *ontology-extended data source* as a tuple $\mathscr{D}_i = <D_i, S_i, O_i>$, where D_i refers to the data contained in the data source, S_i is the schema of the data source, and O_i is the ontology associated with D_i. In addition, the following condition also needs to be satisfied: $D_i \subseteq \Gamma_1^i \times \cdots \times \Gamma_m^i$, which means that the set of values each attribute A_j^i can have is determined by its type Γ_j^i defined in the ontology O_i.

2.2 User Perspective and Ontology Mapping

Suppose $\mathscr{D}_1, \cdots, \mathscr{D}_n$ be an ordered set of ontology-extended data sources and U an user who wants to query $D_1, \cdots, D_n$ semantically heterogeneous data

sources. A user perspective is given by the user ontology O_U and a set of interoperation constraints that define the correspondences between the terms and concepts in $O_1, \cdots, O_n$ respectively, with the user ontology O_U. These interoperation constraints can take one of the following forms [5]: $x : O_i \sqsubseteq y : O_U$ (x is semantically subsumed by y), $x : O_i \sqsupseteq y : O_U$ (x semantically subsumes y), $x : O_i \equiv y : O_U$ (x is semantically equivalent to y), $x : O_i \neq y : O_U$ (x is semantically incompatible to y), $x : O_i \approx y : O_U$ (x is semantically compatible with y). As shown in [9], the set of mappings can be semi-automatically inferred from the set of interoperation constraints. INDUS also provides a graphical user interface to specify the interoperation constraints [6].

2.3 Knowledge Acquisition algorithms

It has been shown in [9] that the functioning of various knowledge acquisition and decision-making algorithms (e.g., classifier learning algorithms) can be reduced to answering queries from distributed data sources by decomposing it into two sub-tasks: *information extraction* and *hypothesis generation*. The information extraction component identifies the required *sufficient statistics*[3] information, whereas, the hypothesis generation component uses this information to generate a predictive model (Figure 2).
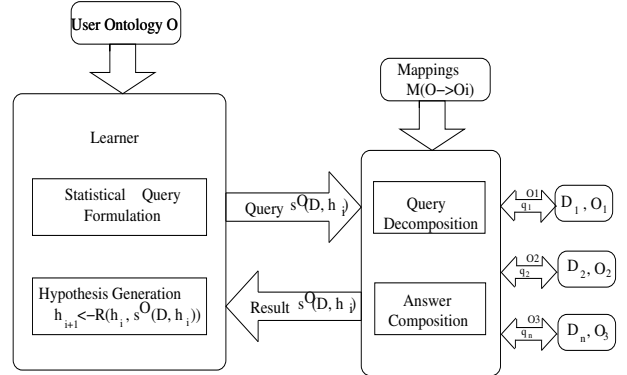

Figure 2. Learning from Distributed, Semantically Heterogeneous Data Sources [9]

The information extraction component typically involves a procedure for determining the sufficient statistics as a *query* and a procedure for answering these queries from the distributed data sources.

---

[2]A *statistic* is simply a function of data and any kind of query that returns such a statistic is called a *statistical query*. Examples of statistic include counts of instances that have specified values from a subset of attributes.

[3]A statistic $s(D)$ is called a sufficient statistic for a parameter θ if $s(D)$ (loosely speaking) provides all the information needed for estimating θ from data D [10]. For example, sample mean is a sufficient statistic for mean of a Gaussian distribution.

The process of answering queries from distributed data requires decomposition of the original query into sub-queries, for which the individual data sources can respond. These responses are then composed into a final answer for the original query. In case of semantically heterogeneous, distributed data sources, the mappings between the user ontology and data source ontologies also need to be applied. Thus, through the means of a query answering engine, this process can be made transparent to the functioning of the knowledge acquisition algorithms, and hence such algorithms can be regarded as *pseudo-users* in INDUS.

Designing models for estimating probabilistic failure indices of power system equipment by capturing the uncertainty relationship between the observations and actual deterioration states is important for representing equipment state in system-level decision algorithms. The procedure for generating such models can be similarly decomposed into information extraction and hypothesis generation components. As a result, such algorithms can be easily connected to INDUS for efficient knowledge acquisition from distributed, semantically heterogeneous data sources. In what follows, we will show how we have used Hidden Markov Models with INDUS for failure rate probability determination for power transformers.

3 HMM for Failure Mode Estimation

The average age of transmission equipment has increased significantly during the past 20 years. As a result, the amount of condition monitoring has also increased, and many utilities are now maintaining extended condition histories. There has also been significant work in developing diagnostics, mainly in the form of rules that we call deterioration functions, used to operate on condition measurements and identify the state of a piece of equipment with respect to a particular failure mode. But there has not been corresponding efforts to transform condition data into a form that can be used in system-level decision tools. Such tools include, for example, maintenance selection and scheduling and transmission reliability evaluation. The standard representation for equipment state in such tools is via a probabilistic failure index such as *failure rate, failure probability,* or *time to failure.* Therefore, to utilize the rich information that is embedded in the increasingly available condition histories, it is necessary to transform the condition histories into such probabilistic failure indices. The limited amount of work towards this end includes [8, 14, 15].

We introduce in this section Hidden Markov Model (HMM), which is very well suited for this task. Al-

though it has been used most heavily in speech processing [24], it has also been used for failure pattern reorganization and condition monitoring using current data [18] and acoustic vibration data [4]. Our application of HMM is extended from application of multi-state Markov models to the same problem [19], which were adapted from models presented in [14]. Markov-based models are desirable because they are inherently suited to modeling multi-state processes such as equipment deterioration. Condition of equipment is divided into states corresponding to intervals of deterioration as computed from deterioration functions operating on condition measurements (Figure 3). IEEE has devel-
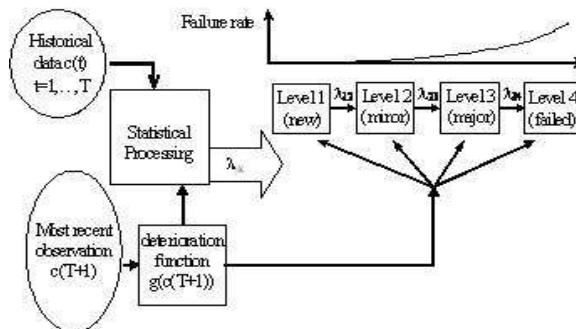

Figure 3. Computing Contingency Probability Reductions

oped a standard to interpret the insulation conditions of oil-immersed power transformer based on Dissolved Gas Analysis (DGA) [1]. This standard classifies transformer conditions into 4 discrete deteriorating states, with the criteria of combustible gases as by-product of insulation deterioration, as listed in Table 1. We set up our Markov model based on this standard.

However, standard Markov model assumes that the deterioration function provides perfect identification of the state. HHM improves on this approach because it accounts for uncertainty in state identification, enabling representation of uncertainties in the mappings between observations and states. HMM's appropriate models for discrete-time, discrete-space dynamical systems governed by a Markov chain, is a statistical model that uses probability measures to represent sequence of observation vectors. It is a composition of two stochastic processes, a *hidden Markov chain*, which accounts for real status of the deterioration, and an *observable process*, which accounts for observation obtained from monitoring and tests. When the component is in a particular state, we characterize the probability that a particular measurement can be generated according to

Status	Dissolved Key Gas Concentration Limits (ppm)							
	H_2	CH_4	C_2H_2	C_2H_4	C_2H_6	CO	CO_2	Total Dissolved Combustible Gas
Condition 1	<100	<120	<35	<50	<65	<350	<2500	<720
Condition 2	101-700	121-400	36-50	51-100	66-100	351-570	2500-4000	721-1920
Condition 3	701-1800	401-1000	51-80	101-200	101-150	571-1400	4001-10000	1921-4630
Condition 4	>1800	>1000	>80	>200	>150	>1400	>10000	>4630

Table 1. Determine Transformer Condition based on DGA [1]

an assumed probability distribution. It is only the outcome, and not the state that is visible to an external observer, and therefore states are 'hidden'.

A HMM is characterized with the following parameters:

1. Markov transition matrix: state transition probabilities A = $\{a_{ij}\}$, $a_{ij} = p(q_{t+1} = j|q_t = i)$, $1 \leq i \leq N$, where q_t denotes the current state.

2. Probability of getting an observation with a symbol under specific state B = $\{b_j(k)\}$, $b_j(k) = p\{o_t = v_k|q_t = j\}$, $1 \leq j \leq N$, $1 \leq k \leq M$, where o_t denotes the current observation.

3. Initial state distribution $\Pi = \{\pi_i\}$, where $\pi_i = p\{q_1 = i\}$, $1 \leq i \leq N$.

This is a learning problem, where we adjust the HMM parameters so that the given set of observations are represented by the model in the sense of maximum likelihood, which means to get the optimal parameter, $\lambda = \{A, B, \Pi\}$, by maximizing the likelihood of observation, $L_{tot} = p(O|\lambda)$. There have been well-developed methods for doing this, like Baum-Welch Algorithm (also known as forward-backward algorithm) [7]. The algorithm includes two parts: 1) Transforming the objective function $p(O|\lambda)$ into a new function, $F(\lambda, \lambda')$, that measures a divergence between the initial model λ and upgraded model λ'; 2) Maximizing the function $F(\lambda, \lambda')$ over λ' to improve λ in the sense of increasing the likelihood $p(O|\lambda)$. The algorithm continues by replacing λ with λ', and repeating the two steps until some stopping criteria is met. In this way, the method is used to fit the test data in the sense of maximum likelihood estimation.

After the HMM transition intensities are determined, the transition probability matrix for the model of Figure 3 can be obtained by Equation (1). The state probability vector gives the probability that a component is in any particular deterioration level at a given time, and is denoted by: $p(hT) = [p_1(hT) \cdot p_2(hT) \cdot p_3(hT) \cdot p_4(hT)]$, where $h = 1, 2, 3, \cdots$, and T is the time increment. If at time $t = 0$, the component

resides in deterioration level 1, then the initial state probability vector is $p(0) = [1\ 0\ 0\ 0]$. The probability of finding the component in any deterioration level at the time hT is then given by $p(hT) = p(0) \cdot \mathbf{P}^h$, where the last number of each probability vector $p(hT)$ corresponds to the probability that the component is in the state of failure before time hT, or the CDF (cumulative density function) of failure. The time to failure may be obtained as first passage times [12].

$$\mathbf{P} = \begin{pmatrix} 1-\lambda_{12} & \lambda_{12} & 0 & 0 \\ 0 & 1-\lambda_{23} & \lambda_{23} & 0 \\ 0 & 0 & 1-\lambda_{34} & \lambda_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

In Section 4.2, we illustrate use of HMM to investigate the failure rate corresponding to the deterioration of oil in transformer, with the data of *dissolved gas analysis* (DGA).

4 System Design and Experimentation

4.1 INDUS Implementation

INDUS comprises of five principle modules (Figure 4): *graphical user interface, ontology & mapping repository, query answering engine, data mining algorithms & code repository* and *data source & wrappers registry*. The modular design of INDUS ensures that each module can be updated and alternative implementation easily explored.

The *graphical user interface* allows the users to interact with the system. It provides an editor [6] for specifying the ontologies and mappings. It also allows the users to register data sources (and their corresponding wrappers) and various data mining algorithms with INDUS. Using the interface, the users can specify queries over distributed, semantically heterogeneous data sources.

The *ontology & mapping repository* stores the various data source ontologies and user-defined ontologies. It also contains the set of mappings between the terms
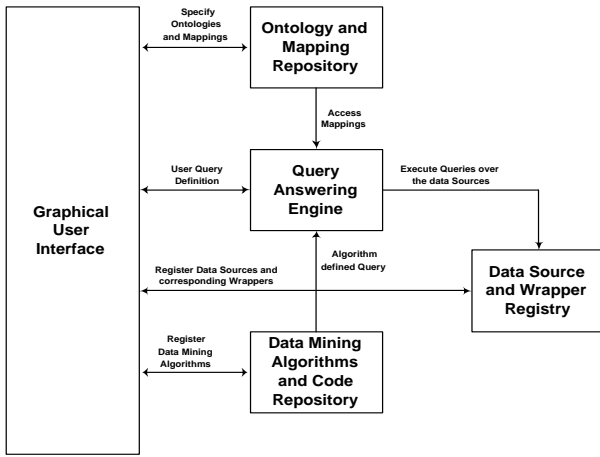
Figure 4. INDUS Implementation Modules

and concepts in the user ontology and data source ontologies. These mappings are accessed during query processing and execution.

The *data source & wrapper registry* allows the users to register various data sources and wrappers with the system. These wrappers provide a set of functions to interact with the individual data sources. Each wrapper is implemented by a Java class. During the registration of the data sources, the users also provide a *capability description* of the data sources. Such descriptions provide information about the structure of the data source (e.g., relational, XML), querying capabilities (e.g., different types of functionalities the data source provides), querying restrictions (e.g., various constraints on the usage of data by external applications), infrastructure (e.g., CPU speed, RAM size of the server hosting the data source) etc. These informations are used during query execution.

The *data mining algorithms & code repository* allows users to register various data mining and knowledge acquisition algorithms. These algorithms act as *pseudo-users* in INDUS. This repository also allows users to store application-specific functionalities that might be used in querying the registered data sources.

Finally, the *query answering engine* accepts a query either from an user or from data mining algorithms (i.e., the information extraction component). This engine acts as a middleware between the users and data sources, and utilizes the functionalities of the data source wrappers for query processing. There are two main aspects of the engine. Firstly, it *translates the user queries* (which are specified using the concepts in the user ontology) into data-source specific queries via the interoperation constraints (or ontology mappings), hence allowing the users to view the data source from their own point of view. Secondly, the engine

adopts a *hybrid query answering* approach, which allows it to choose to perform some query execution at the data source server, and some portion of the execution at the client location. The rationale behind this design choice is that, this approach allows the engine to decide whether to ship executable code (for query answering) to the data source server location, or ship raw data to the client location for local processing based on the dynamics of the query and various querying capabilities of the data source (as specified in the data source description). The engine comprises of 4 sub-components: *Query Decomposition, Query Translation, Query Execution* and *Answer Composition*. Upon receiving a query Q (based on concepts in user ontology O_U) from the user/application, the query decomposition component identifies the data sources, $D_1, \cdots, D_n$, that need to be queried, and decomposes the original query into sub-queries, $Q_{D_1}, \cdots, Q_{D_n}$, that are sent to the query translation component. For each sub-query, Q_{D_i}, received by this component, it is translated (or re-written) in terms of the concepts specified in the data source ontology, O_i. The translated sub-query is then sent to the query execution component which enumerates alternate plans for processing the query, and executes the one which is most efficient. Finally, the result of the sub-query is sent to the answer composition component. This component composes the partial answers (i.e., the results of all the sub-queries) into a final answer for the original query Q, and sends it back to the user.

In what follows, we demonstrate an application of INDUS for failure rate estimation using condition monitoring data.

4.2 Transformer Failure Rate Estimation based on Condition Monitoring Data

Condition monitoring is an important method in maintenance asset management of components in the transmission system. Relative to the conservative time-based maintenance, which utilizes the fixed maintenance intervals, condition monitoring based maintenance only triggers maintenance when an incipient failure is identified with the information characterizing the equipment conditions. Thus, it typically extends the interval between successive maintenances and therefore incurs less cost. However, it requires a significant amount of infrastructure investment (e.g. sensors, diagnostic technology, communication channels, data repositories and processing software) to measure, communicate, store and utilize the necessary information characterizing the state of the equipment. There have been many condition monitoring techniques cor-

responding to different failure modes of transformers, including dissolved gas analysis (DGA) results on main tank oil (insulation deterioration, deterioration of cooling system, oil pump failure) and load tap changer oil (oil dielectric weakening), thermography testing (magnetic circuit overheating, bushing overheating), ultrasonic testing (oil pump failure), partial discharge testing (magnetic circuit overheating), winding and oil temperature (deterioration of cooling system), etc. In this paper, we use DGA data to estimate the failure rate of deterioration of insulation oil in transformer.

Mineral oils are used in the transformer tank for insulation and also, as a media for heat transfer. The oils are mixtures of many different hydrocarbon molecules, which decompose under high thermal and electrical stress within the transformer during the period of service. The critical changes are the breaking of carbon-hydrogen and carbon-carbon bonds, as a result of which different gases are formed due to the presence of individual hydrocarbons, and the distribution of energy and temperature in the neighborhood of the fault. IEEE has provided the interpretation of the gases generated in the transformer and corresponding standards for evaluating the condition of transformer oil insulation based on DGA results [1]. In transformer oil analysis, $TDCG$ (*total dissolved combustible gases*, which is the summation of concentration of hydrogen, ethylene, acetylene, methane, ethane and carbon-monoxide) has been utilized as an important indicator of condition of transformer oil and is used as a principle factor for determining the operating procedures for inspection and maintenance intervals [1].

Sample Date	H_2	C_2H_4	C_2H_2	CH_4	C_2H_6	CO	$TDCG$
15-Sep-95	3	9	0	19	4	539	574
18-Sep-96	0	13	0	20	9	467	509
09-May-97	0	9	0	30	3	578	620
27-Aug-98	26	22	0	54	10	942	1054
12-Apr-99	21	28	0	60	6	731	846
10-Sep-02	305	691	0	648	192	657	2493
15-Oct-02	569	1703	7	1364	451	552	4646
22-Oct-02	573	1965	6	1637	520	643	5344
28-Oct-02	557	2002	7	1616	535	599	5316
10-Dec-02	1	22	0	7	6	5	41

Table 2. DGA Test Data for a Transformer

Such information can be gathered from the condition monitoring data sources using INDUS. Specifically, we determine a transformer we want to analyze, and send a query request to INDUS for accumulation of TDCG information by using the ID (an unique identifier) of the transformer and the data period we want to examine. Table 2 shows results to a query for gathering DGA analysis information for one transformer between two maintenance periods of oil filtering, which is the maintenance activity corresponding to the failure mode oil deterioration. As can be seen from Table 2, there is a sharp decline in the concentration of various gases in the last record. This indicates a maintenance activity, confirmed by the maintenance history data, and as a result, we use all the records, *except* the last one, to simulate the deterioration process using a HMM. We achieve this by incorporating the generation of HMM with INDUS. This would allow the HMM (i.e., pseudo-user) to ask queries for gathering DGA analysis information over physically distributed, autonomous, and semantically heterogeneous data sources. Once the relevant information is extracted, the algorithm can estimate transition intensities for the Markov model (Table 3).

$Transition Rate$	1	2	3	4
λ i, i	0.9917	0.9915	0.9807	1.000
λ i, i+1	0.0083	0.0085	0.0193	0.000

Table 3. Estimated Transition Intensities for Markov Model

To validate our results, we compare the observations with the results obtained from HMM. In Table 4, *Es* is the status of the components with observation data (interpreted using a deterioration function based the IEEE Standards [1]), and *Eu* is the forecasted states that the component will be at different time, with our HMM model. We observe that they match very well, suggesting that the HMM can be effectively used to simulate the deterioration process.

Time (week)	1	54	87	155	187	366	371	372	372
Es	1	1	1	2	2	3	4	4	4
Eu	1	1	1	2	2	3	4	4	4

Table 4. Comparison of Observation and Forecast

The probability that we need to calculate is *failure rate*, or hazard function [22], which is the instantaneous probability of the component to fail during the period of $[(h+1)T, hT]$, given the condition that it survives to time hT. This probability can be calculated as follows:

$$Pr(hT \leq x \leq (h+1)T \mid x > hT) = \frac{p((h+1)T) - p(hT)}{1 - p(hT)}$$

8

Figure 5 shows the distribution of calculated failure rate vs. time. This graph can be used to calculate the change of failure probability after the maintenance. In Table 2, the last records show that maintenance was performed 377 weeks after the first record (the date of previous maintenance). This information can then be used to check the failure probability for the time period without maintenance. For example, from Figure 5, we can determine $Pr(377) = 0.004354$. Since the record after maintenance shows that the oil is in very good condition, we can infer that maintenance renews the oil, as a result of which, the failure probability returns to 0. Thus, we can calculate ΔPr, the change of failure probability after maintenance, will be 0.004354.
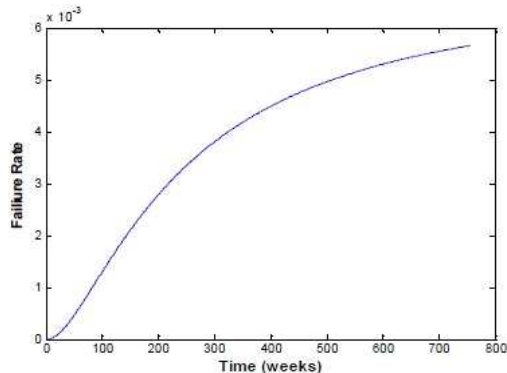

Figure 5. Failure Rate of Transformer Oil Deterioration

We can also calculate the expected time to failure with the results from HMM (Table 5). It is captured by computing *first passage times*, which is the expected value of the amount of time it will take to transit from a given state j to another state i, under the assumption that the process begins in state j. From this computation, then, we may estimate the remaining life of the component. We utilize the method introduced in [2,12] to calculate the first passage time to failure as follows:

$$T_f = p(0) \times T \times (1 - Pr(T))^{-1}$$

where, $p(0)$ is the initial state of the Markov process, T is the time unit of each step, T_f is the vector of time to failure from different states, $Pr(T)$ is a partition of the transition matrix $\mathbf{P}$ corresponding to the non-failure states. Table 5 gives the results for components in each state, the average time to next state, and the estimated time to failure.

State	1	2	3
Time to next state (weeks)	120.5	155.4	91.9
Time to failure (weeks)	367.8	247.3	91.9

Table 5. First Passage Time for each State

5 Summary and Discussion

This paper addresses a highly complex dynamic data-driven decision problem associated with a critical national infrastructure - *asset management for the electric power system.* Solution to this problem involves six main issues: 1.) Sensing and diagnostics; 2.) Data accessibility, communication, and integration; 3.) Data transformation; 4.) System simulation across multiple decision horizons; 5.) Decision making; and, 6.) Information valuation and sensor deployment or re-deployment.

There are four different kinds of decisions to be made. Operational decisions are made within the hour to week time frame and require trading off risk associated with potential equipment failure with the short-term economics of generation dispatch. Maintenance decisions are made within the week to year time frame and require allocating financial and human resources to maximize benefits in terms of operational reliability and equipment life. Planning decisions are made withing the 1-10 year time period and require determining the necessary and most effective capital improvements in terms of facility investments to continue supply of the growing load from expanding energy resources. Each of these decisions affect others, and so the capability to capture the interaction of different policies in one decision-horizon with those of another decision-horizon is essential. Fourth, it is through the simulation and inter-related first three decision problems that one may be able to determine where additional information would be valuable. This information valuation problem, #6 on the above list, determines where to deploy new sensors and associated infrastructure to collect additional information. In a real sense, then, this dynamic data-driven decision problem is *closed*, i.e., it feedbacks on itself.

In this paper, we have addressed two of the issues listed above: #2 (data integration) and item #3 (data transformation). The data federation approach of the INDUS platform provides a rich alternative to the data warehousing approach used in industry today, with important benefits being that data need not be moved except at the instant it is needed, and as a result, simulation models are always making use of the very latest equipment condition measurements. The HMM

provides an essential bridge between condition data and the probabilistic failure indices required by the system simulation tools of issue #4 above. It is quite natural that the data integration tools would interface closely with the data transformation applications, as illustrated by the design presented in this paper. We intend to continue expanding this prototype to include application software associated with the other issues listed above.

References

[1] IEEE Standards C57, IEEE Guide for the Interpretation of Gases Generated in Oil-Immersed Transformers Pages 104-1991.

[2] G. Anders. *Probability Concepts in Electric Power Systems*. John Wiley, 1990.

[3] Y. Arens, C. Chin, C. Hsu, and C. Knoblock. Retrieving and Integrating Data from Multiple Information Sources. *International Journal on Intelligent and Cooperative Information Systems*, 2(2):127–158, 1993.

[4] L. Atlas, M. Ostendorf, and G. Bernard. Hidden Markov Models for Monitoring Machining Tool-Wear. In *IEEE Intl. Conference on Acoustics, Speech, and Signal Processing*, 2000.

[5] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. Cambridge University Press, 2003.

[6] J. Bao and V. Honavar. Collaborative Ontology Building with Wiki@nt. In *3rd Intl. Workshop on Evaluation of Ontology Based Tools at Intl. Semantic Web Conference*, 2004.

[7] L. Baum and T. Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *Annals of Mathematical Statistics*, 37:1554–1563, 1966.

[8] L. Bertling, R. Allan, and R. Eriksson. A Reliability-Centered Asset Maintenance Method for Assessing the Impact of Maintenance in Power Distribution Systems. *IEEE Transactions on Power Systems*, (1):75–82, 2005.

[9] D. Caragea, J. Pathak, and V. Honavar. Learning Classifiers from Semantically Heterogeneous Data Sources. In *3rd Intl. Conference on Ontologies, DataBases, and Applications of Semantics for Large Scale Information Systems*, 2004.

[10] G. Casella and R. Berger. *Statistical Inference*. Duxbury Press, Belmont, CA, 2001.

[11] S. Ceri, S. Navathe, and G. Wiederhold. Distribution Design of Logical Database Schemas. *IEEE Transactions on Software Engineering*, 9(3):487–504, 1983.

[12] A. L. da Silva and J. Endrenyi. Application of First Passage Times in the Markov Representation of Electric Power Systems. In *4th Intl. Conference on Probabilistic Methods Applied to Power Systems*, 1994.

[13] D. Draper, A. Y. Halevy, and D. S. Weld. The Nimble XML Data Integration System. In *Intl. Conference on Data Engineering*, pages 155–160, 2001.

[14] J. Endrenyi, G. Anders, and G. LeitedaSilva. Probabilistic Evaluation of the Effect of Maintenance on Reliability - An Application. *IEEE Transactions on Power System*, 13(2):576–583, 1998.

[15] L. Fangxing and R. Brown. A Cost-Effective Approach of Prioritizing Distribution Maintenance based on System Reliability. *IEEE Transactions on Power Delivery*, (1):439–441, 2004.

[16] H. Garcia-Molina and *et al.* The TSIMMIS Approach to Mediation: Data Models and Languages. *Journal of Intelligent Information Systems*, 8(2), 1997.

[17] P. Gruber and J. Wills. *Handbook of Convex Geometry*. Elsevier Science Publishers B.V., 1993.

[18] E. Hatzipantelis and J. Penman. The use of Hidden Markov Models for Condition Monitoring Electrical Machines. In *6th Intl. Conference on Electrical Machines and Drives*, 1993.

[19] Y. Jiang, Z. Zhong, J. McCalley, and T. V. Voorhis. Risk-based Maintenance Optimization for Transmission Equipment. In *Proc. of 12th Annual Substations Equipment Diagnostics Conference*, 2004.

[20] A. Levy. The Information Manifold Approach to Data Integration. *IEEE Intelligent Systems*, 13, 1998.

[21] J. Lu, G. Moerkotte, J. Schue, and V. Subrahmanian. Efficient Maintenance of Materialized Mediated Views. In *ACM SIGMOD Conference on Management of Data*, San Jose, CA, 1995.

[22] W. Q. Meeker and L. A. Escobar. *Statistical Methods for Reliability Data*. Wiley and Sons, 1998.

[23] E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-Existing Ontologies. *Intl. Journal of Parallel and Distributed Databases*, 8(2):232–271, 2000.

[24] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. pages 267–296, 1990.

[25] L. R. Rabiner and B. H. Juang. An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, pages 4–15, January 1986.

[26] J. Reinoso-Castillo, A. Silvescu, D. Caragea, J. Pathak, and V. Honavar. Information Extraction and Integration from Heterogeneous, Distributed, Autonomous Information Sources: A Federated, Query-Centric Approach. In *IEEE Intl. Conference on Information Integration and Reuse*, 2003.