RC22682 (W0212-109) December 16, 2002
Computer Science

IBM Research Report

GAP: A General Approach to Quantitative Diagnosis of Performance Problems

Joseph L Hellerstein
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Delhi - Haifa - India - T. J. Watson - Tokyo - Zurich

GAP: A General Approach to Quantitative Diagnosis of Performance Problems

Joseph L. Hellerstein
IBM Research Division
T.J. Watson Research Center
Yorktown Heights, NY 10598

Key words: model-based diagnosis, problem determination, quantitative decision support

Abstract

Quantitative performance diagnosis (QPD) provides explanations that quantify the impact of problem causes. An example of such an explanation is Increased web server traffic accounts for 90% of the increase in LAN utilization, which in turn accounts for 20% of the increase in web response times. This paper describes GAP, a general approach to quantitative performance diagnosis. GAP has two parts: (1) an algorithm for computing quantitative performance diagnoses and (2) a framework for constructing diagnostic techniques that provides the basis for quantifications produced by the algorithm. The GAP algorithm makes use of a measurement navigation graph (MNG), a directed acyclic graph whose nodes are measurement variables and whose arcs have weights that quantify the effect of child variables (e.g., LAN utilization) on parent variables (e.g., response time). Various properties of the algorithm are established, especially that its quantification of explanations can be interpreted as fractional contributions to the performance problem. Arc weights are computed by diagnostic techniques. A framework for developing diagnostic techniques is described that consists of (a) the choice of statistic (e.g., mean, variance) to aggregate problem values and (b) the estimator of the statistic. The framework is applied to existing diagnostic techniques to assess their effectiveness and is used to construct a new diagnostic technique for a performance problems in a production computing systems. It is also used to show that for uniform magnitude performance problems (e.g., a step), the standard deviation is preferred to the mean if the problem data have a coefficient of variation no larger than $\sqrt{(1-f)/f}$, where f is the fraction of the data containing the performance problem.

1 Introduction

The advent of distributed systems has produced numerous productivity and price benefits. However, distributed systems have also contributed to the soaring cost of operations and management. According to industry analysts, these costs account for 60% to 80% of the cost of owning network-connected computers (e.g., [12]).

A critical element of operations and management is addressing performance problems, such as long response times in eCommerce systems and low throughputs for nightly database updates. Such considerations require mechanisms for detecting, diagnosing, and resolving performance problems. Detection involves using one or more measurement variables to sense when a problem occurs, such as using on-line change-point detection algorithms to sense changes in web response times. Diagnosis isolates problems to specific components, such as attributing large web response times to excessive LAN utilizations. Resolution involves selecting and implementing actions that eliminate the problem, such as increasing LAN capacity or reducing LAN traffic.

This paper addresses **quantitative performance diagnosis (QPD)**. Quantitative performance diagnosis consists of a set of explanations and a quantification of their importance. Ideally, this quantification takes the form of fractional contributions to the performance problem, such as:

> The 30% increase in web server traffic accounts for 90% of the increase in LAN utilization, which in turn accounts for 20% of the increase in web response times.

Two benefits accrue from providing quantitative diagnoses. First, since many factors affect performance, analysts and administrators can focus on those that contribute most to the performance problem. Second, quantitative information is often needed to specify the actions for resolving performance problems. Examples of such actions are: (1) by how much must the LAN capacity be increased and (2) by how much LAN traffic must be reduced.

Numerous applications have been developed for diagnosing performance problems in computer and communications systems (e.g., see [8] for an overview). The most common approach employs hand-crafted if-then rules (e.g., [3], [9], [11], [13], [16]) that produce *qualitative* diagnoses, such as Large response times are due to excessive LAN utilization. This approach results in "brittle" systems that fare poorly if configurations or

device characteristics change since this knowledge is typically embedded in the rules. Brittleness is reduced by externalizing knowledge of the system being diagnosed. In fault diagnosis, this is achieved by taking as input external models of the system, such as failure modes (e.g., [24]), qualitative models of system behavior (e.g., [10], [21]) and fault trees (e.g., [17], [29]). More recently, techniques such has hidden markov models have been used to address uncertainties in diagnosis [31]. In performance diagnosis, generality is typically achieved by using a directed graph to express dependencies between measurement variables (e.g., [5], [23], [4]). Herein, this is referred to this as a **measurement navigation graph (MNG)**. Unfortunately, a MNG by itself lacks the information necessary for QPD. Thus, some have approached QPD by using custom-tailored analytic models (e.g., [7], [25]), but this creates the same brittleness as with hand-crafted rules. Another approach is to externalize quantitative relationships between measurement variables. Several approaches have been suggested (e.g., [13], [14], [19]), but none have proposed a systematic algorithm. Further, the approaches suggested appear to assume that the MNG is a tree, a structure that precludes handling multi-parent dependencies that arise when there are shared resources.

Another related area is diagnosis of electrical circuits. Examples here include: a combination of model-based and case-based approaches to diagnosing electrical circuits [28]; statistical approaches to determining the cause of path delays in electrical circuits [26]; and minimizing the number of measurements made to diagnose an analog circuit [1]. All of these approaches provide quanitifications. However, the focus is much different than in this paper. The first two assume that either the results of past diagnoses are available or repeated measurements can be made in a controlled manner. Neither assumption applies in the situations considered in this paper. The third focuses on minimizing the number of tests required, not addressing the post-mortem analysis of a problem that occurred. Such post-mortem analyses are of interest in domains other than computing systems. Examples include asset management problems in power plants [27], corporate financial performance [19], and resource utilizations in hospitals [8].

This paper describes **GAP**, a **g**eneral **a**pproach to quantitative **p**erformance diagnosis. The paper describes the GAP algorithm for quantitative performance diagnosis and applies it to a performance problem in a production computing system. Various properties of the algorithm are established, especially that its quantification of explanations can be interpreted as fractional contributions to the performance problem. GAP operates by using **diagnostic techniques** that assign weights to arcs in the MNG so as to determine the best explanations of performance problems. The paper describes a framework for constructing diagnostic

techniques based on the statistic used to aggregate data and the estimator of this statistic. Among other things, an analysis is presented of the choice of statistic between using the mean and the standard deviation in the specification of a diagnostic technique. Specifically, it is shown that for uniform magnitude performance problems (e.g., a step), the standard deviation is preferred to the mean if the problem data have a coefficient of variation no larger than $\sqrt{(1-f)/f}$, where f is the fraction of the data containing the performance problem.

The remainder of this paper is organized as follows. Section 2 discusses the GAP algorithm. Section 3 describes the GAP framework for constructing diagnostic techniques. Section 4 discusses and extends the results presented. Conclusions are presented in Section 5.

2 Algorithm for Quantitative Performance Diagnosis

This section presents the GAP algorithm for quantitative performance diagnosis. A running example is introduced to aid in this presentation and facilitate discussion in the remainder of the paper. The GAP algorithm is described in detail, and it is applied to the running example.

The running example illustrates the nature of performance problems and their diagnosis. Considered is a production computer installation at a large telecommunications company that uses a mainframe computer to provide interactive time sharing to several hundred engineers, scientists, and secretaries. Quality of service is characterized by average response time. Fig. 1 plots this metric versus time in thirty second intervals for a day during which several performance problems occurred. From 8:00 AM through 10:00 AM, there are large response times approximately every 10 minutes as a result of a poorly designed, CPU-intensive application. From approximately 10:45 through 11:30 AM, there is an abrupt increase in response time due to a software error in the paging subsystem. From 2:30 through 4:00 PM, there is a gradual increase in response time due to a memory leak.

Fig. 2 displays a queueing diagram for the running example. In this diagram, end-users initiate requests (e.g., by pressing the *Enter* key), which result in requests to the CPU subsystem. Following this, there may be requests for the paging and/or user IO subsystems, which in turn may be followed by other requests for the CPU subsystem. In the figure, there are annotations that indicate measurement variables for response

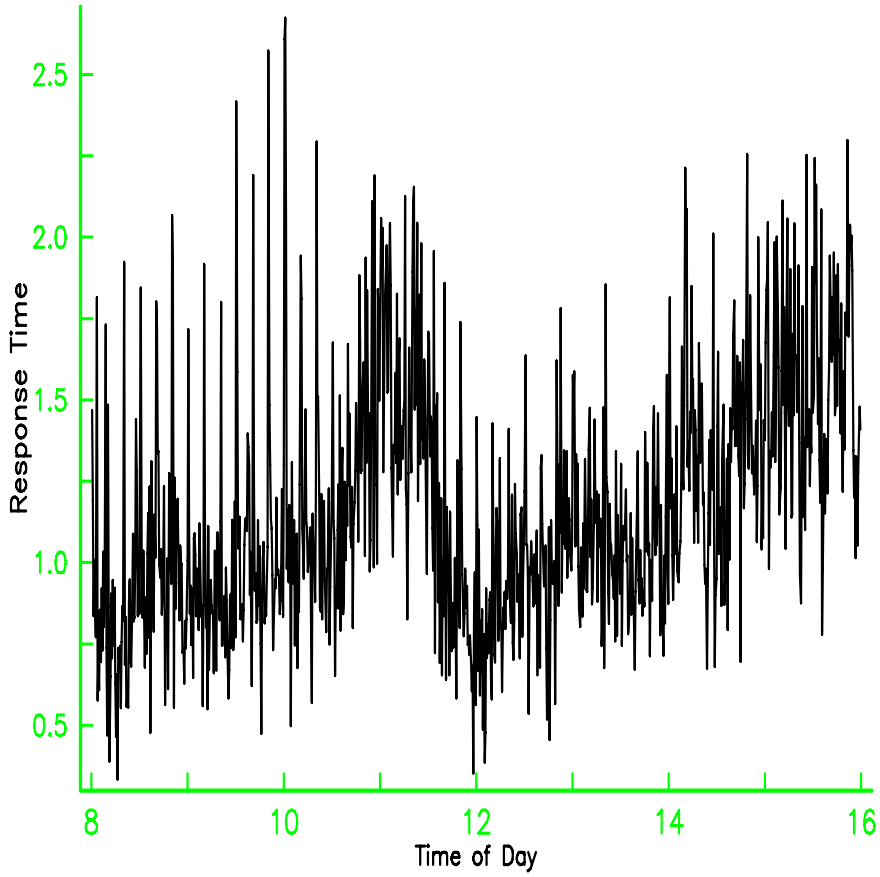

Figure 1: Response Times for Performance Problems in a Production Computer System

---

times, service times, and request rates at each queue. Response time variables begin with RT, such as $RTTOT$ (with units of time). Service time variables have the prefix ST (with units of time), and transaction rate variables start with TR (with units of customers per time). Further, let n_{CPU}, n_{IO}, and n_{PAGE} denote the number of subsystem visits per user request (a dimensionless quantity). The quantity $n_i RTi$ (for $i \in \{CPU, IO, PAGE\}$) is the sojourn time (with units of time) for subsystem i, which is denoted by using the prefix SJ (e.g., $SJCPU$). So,

$$\begin{aligned} RTTOT &= (n_{CPU})(RTCPU) + (n_{IO})(RTIO) + (n_{PAGE})(RTPAGE) + RTOTHER \\ &= SJCPU + SJIO + SJPAGE + RTOTHER, \end{aligned} \tag{1}$$

where $RTOTHER$ is included to handle time not otherwise accounted for.

It turns out that service times for user IO and paging are actually response times for another queueing network, the IO subsystem. This network is depicted in Fig. 3. The forward flow in this figure is from

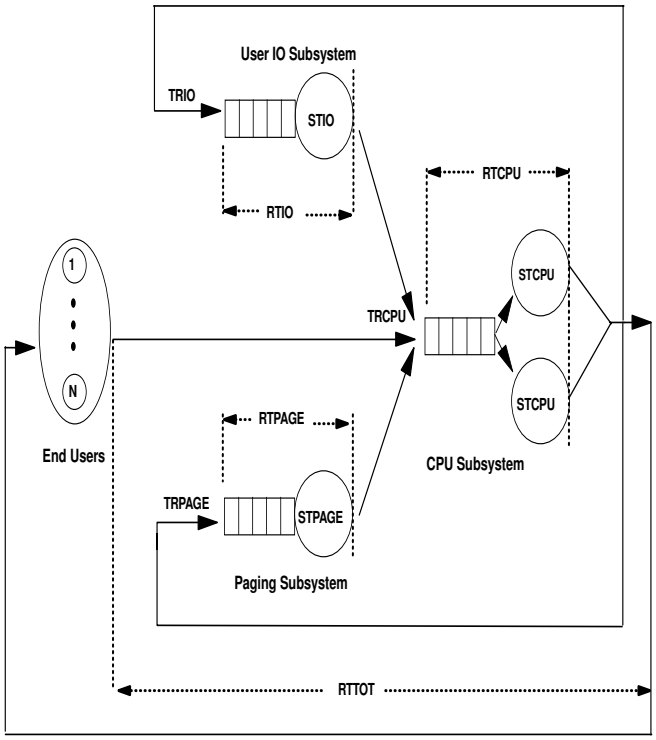

Figure 2: Queueing Diagram for System in Running Example

---

channels (which control interactions between memory and IO devices) to control units (which handle device selection) to the disks. There is also a backward flow that reverses the foregoing. As in Fig. 2, there are annotations for variable names that use the RT, ST, and TR prefixes.

The GAP algorithm operates on a **measurement navigation graph (MNG)**, a directed acyclic graph in which nodes are measurement variables and arcs reflect functional relationships between parent and child variables. Fig. 4 displays a MNG for the system in the running example. Such diagrams are constructed from equations such as Eq. (1) and queueing diagrams such as Fig. 2 and Fig. 3. For example, in Eq. (1), $RTTOT$ appears on the left-hand side and $SJCPU$, $SJIO$, $SJPAGE$, $RTOTHER$ are on the right-hand side. So, the MNG has arcs from $RTTOT$ to $SJCPU$, $SJIO$, $SJPAGE$, and $RTOTHER$. From the queueing diagram in Fig. 2, time spent in the CPU subsystem is determined by $STCPU$ (service times at the CPU) and $TRCPU$ (arrival rates at the CPU). Thus, the MNG has arcs from $SJCPU$ to $STCPU$ and

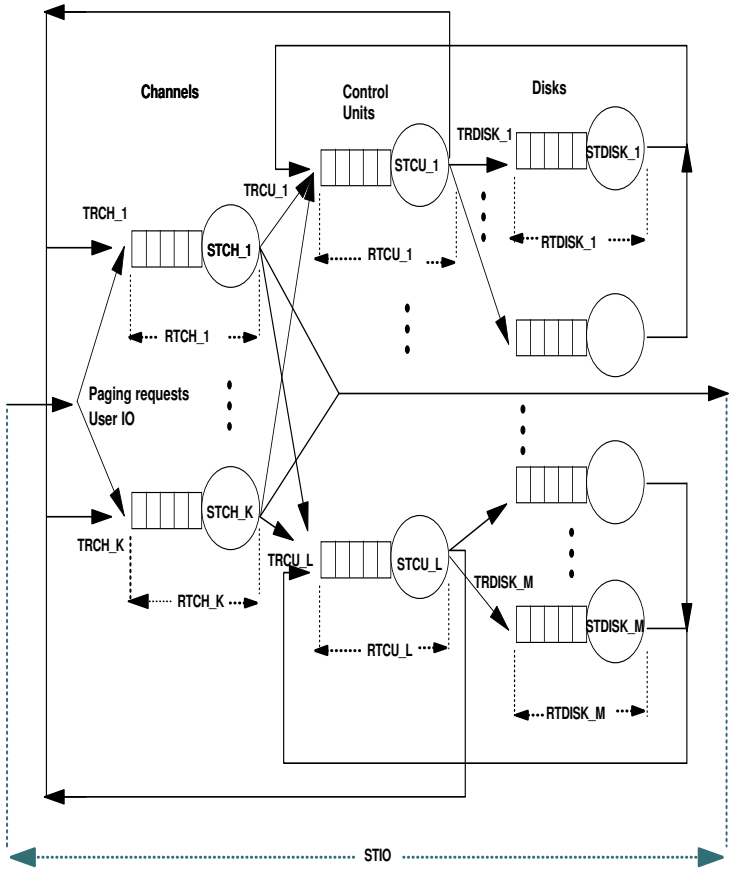

Figure 3: Queueing Diagram for I/O Subsystem in Running Example

---

from $SJCPU$ to $TRCPU$. The remaining nodes and arcs in the MNG are obtained in a similar manner.

Note that Fig. 4 is not a tree since $STIO$ has two parents. This is a consequence of the I/O subsystem being shared between the paging and user input/output subsystems. As a result, the latter two subsystems have $STIO$ as their service times $STIO$, which is a response time in Fig. 3.

The foregoing suggests that it may be common to have MNGs that are not trees as a result of shared resources in the system being analyzed. For example, consider an eCommerce system. It is common for multiple clients to share a server process (e.g., for database access). Further, server processes often share operating system facilities, such as database access and network communication.

Constructing a MNG requires knowledge of the system being diagnosed. In some cases, this knowledge is already available in configuration files and other sources. Indeed, if these files are properly structured, it

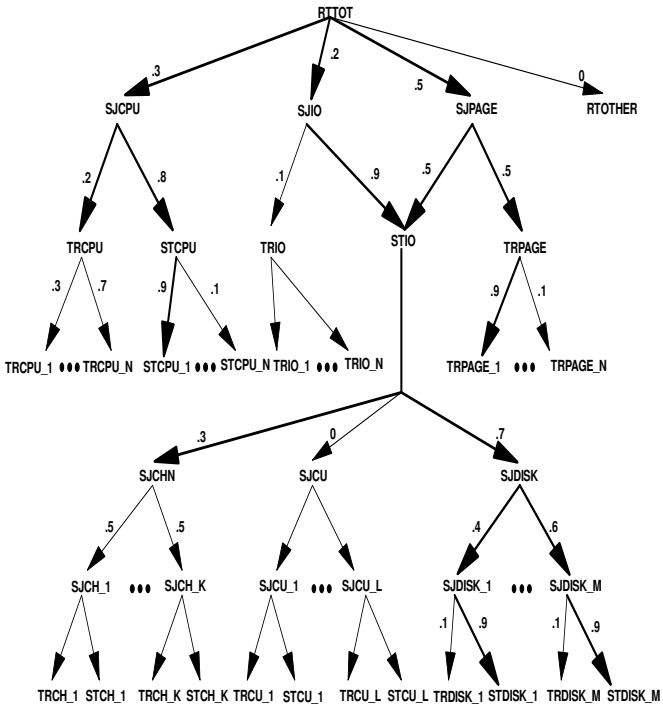

Figure 4: Measurement Navigation Graph (MNG) for Running Example. The arc weights indicate the fractional contribution of child variables to problems detected in the parent variable.

---

may be possible to automate MNG construction [6]. For the most part, however, MNG construction requires the involvement of an expert analyst. However, once an MNG is constructed, it can be used repeatedly until the system or the measurement sources change.

To describe the details of the GAP algorithm for QPD, some notation is required. Let x_i denote a measurement variable, where $0 \leq i \leq N$. x_0 is the variable from which diagnosis begins, which is referred to as the **detection variable**. A MNG is denoted by $G = (V, A)$. V is a set of measurement variables that are vertices in G. A is a set of arcs between measurement variables. $V_L \subseteq V$ is the set of leaves in G. The children of x_i are specified by $chld(x_i) = \{x_j \mid (x_i, x_j) \in A\}$. If x_i is a parent variable, then there is a (possibly unknown) function g_i such that $x_i(t) = g_i(x_{i_1}(t), \cdots, x_{N_i}(t))$, where $x_k(t)$ is the value of x_k at time t. (Note that if there are dynamics in the function, then lags can be incorporated by having additional

variables.) A path from x_i to x_j is defined recursively as:

$$P_{i,j} = \{(x_i, x_k)\} \bigcup P_{k,j},$$

where: $(x_i, x_k) \in A$, and $P_{k,j}$ is a path from x_k to x_j (or $\emptyset$ if $k = j$). With some abuse of notation, $x_k \in P$ indicates that P traverses x_k. $\mathcal{P}_{i,j}$ is the set of all paths from x_i to x_j. The descendents of x_i are those nodes that can be reached from x_i. That is,

$$dcnd(x_i) = \{x_j \mid \mathcal{P}_{i,j} \neq \emptyset\}.$$

A second kind of input to the GAP algorithm are **diagnostic techniques**. Diagnostic techniques quantify the contribution of child variables to performance problems manifested in their parents. For example, consider a diagnostic technique that quantifies the effect on $RTTOT$ due to performance problems manifested in $SJCPU$, $SJIO$, $SJPAGE$, and/or $RTOTHER$. Let $x_0, \cdots, x_4$ correspond to these five variables. Then, one diagnostic technique is to compute arc weights as x_i/x_0, $1 \leq i \leq 4$. Section 4 provides a framework for constructing diagnostic techniques, and evaluates existing approaches.

Arc weights should indicate the *fractional contribution* of child variables to performance problems in the parent. Hence, it is desirable to have **normalized arc weights**. By normalized is meant that: (a) $0 \leq w_{i,j} \leq 1$ and (b) $\sum_j w_{i,j} = 1$. For example, in Fig. 4, all arc weights are normalized.

There are two kinds of outputs from the GAP algorithm. The first are **explanations**, paths in the MNG that explain part of the performance problem. For example, in Fig. 4, one explanation is the path $RTTOT \rightarrow SJCPU \rightarrow TRCPU$. Second, associated with each explanation is a quantification of its importance, or **explanatory power**. A larger explanatory power indicates an explanation that accounts for more of the performance problem in the detection variable (x_0).

Explanatory power is computed as the product of the arc weights in the path for the explanation. Let u_P be the explanatory power for path P. Then,

$$u_P = \prod_{(x_i,x_j)\in P} w_{i,j}.$$

In Fig. 4, the explanatory power of the path $RTTOT \rightarrow SJCPU \rightarrow TRCPU$ is .06.

1. $\mathcal{PU} = \emptyset$; $\mathcal{P} = \emptyset$ /* Initializations */
2. Do for $x_j \in chld(x_0)$ in G /* Form initial paths */
   (a) $P = \{(x_0, x_j)\}$ and compute $w_{0,j}$
   (b) $u_P = w_{0,j}$
   (c) If $u_P \geq THR_{path}$, then $\mathcal{P} = \mathcal{P} \bigcup \{P\}$
3. Do until $\mathcal{P} = \emptyset$ /* Extend the paths */
   (a) Select $P \in \mathcal{P}$ and remove it.
   (b) x_i = last node in P
   (c) If $chld(x_i) = \emptyset$, then $\mathcal{PU} = \mathcal{PU} \bigcup \{(P, u_P)\}$
   (d) For $x_j \in chld(x_i)$
      i. $P' = P \bigcup \{(x_i, x_j)\}$ and compute $w_{i,j}$
      ii. $u_{P'} = (u_P)(w_{i,j})$
      iii. If $u_{P'} \geq THR_{path}$, then $\mathcal{P} = \mathcal{P} \bigcup \{P'\}$
   (e) If no path was added to $\mathcal{P}$ in Step 3(d), then $\mathcal{PU} = \mathcal{PU} \bigcup \{(P, u_P)\}$
4. Return $\mathcal{PU}$

Figure 5: Steps in the GAP Algorithm for Quantitative Performance Diagnosis

Path	u_P
$RTTOT \rightarrow SJCPU \rightarrow STCPU \rightarrow STCPU\_1$	.22
$RTTOT \rightarrow SJCPU \rightarrow TRCPU$	.06
$RTTOT \rightarrow SJIO \rightarrow STIO \rightarrow SJCHN$	.05
$RTTOT \rightarrow SJIO \rightarrow STIO \rightarrow SJDISK \rightarrow SJDISK\_1$	.05
$RTTOT \rightarrow SJIO \rightarrow STIO \rightarrow SJDISK \rightarrow SJDISK\_M \rightarrow STDISK\_M$	.07
$RTTOT \rightarrow SJPAGE \rightarrow STIO \rightarrow SJCHN$	.07
$RTTOT \rightarrow SJPAGE \rightarrow STIO \rightarrow SJDISK \rightarrow SJDISK\_1 \rightarrow STDISK\_1$	.06
$RTTOT \rightarrow SJPAGE \rightarrow STIO \rightarrow SJDISK \rightarrow SJDISK\_M \rightarrow STDISK\_M$	.09
$RTTOT \rightarrow SJPAGE \rightarrow TRPAGE \rightarrow TRPAGE\_1$	.23
Total explained	.91

Figure 6: Example of Output From Diagnosis Algorithm

Fig. 5 displays the steps in GAP algorithm. The algorithm operates by extending paths from the detection variable. A path is extended until either: (a) its explanatory power is too small or (b) a leaf node is reached. Step 1 of the algorithm initializes $\mathcal{PU}$. Step 2 constructs an initial set of paths to be processed. These paths are placed into the set $\mathcal{P}$. Step 3 processes the paths in $\mathcal{P}$. In step 3(a), one path is selected and removed; this path is denoted by P. If P terminates in a leaf, it is added to $\mathcal{PU}$ in Step 3(c). (With some abuse of notation, this is indicated by $P \in \mathcal{PU}$.) Otherwise, all one-node extensions to P are considered. If none of these extensions has a sufficiently large explanatory power, then P is added to $\mathcal{PU}$. Otherwise, the new paths are added to $\mathcal{P}$. Step 3 continues until $\mathcal{P} = \emptyset$. In Step 4, $\mathcal{PU}$ is returned.

Now consider the operation of the GAP algorithm on the running example. Let Fig. 4 be the MNG annotated with the arc weights produced by one or more diagnostic techniques. Further, let $x_0 = RTTOT$ and $THR_{path} = .05$. The heavy lines in Fig. 4 indicate the arcs traversed. Note that no arc is traversed from $TRCPU$ to its children since the resulting path from $RTTOT$ would have an explanatory power less than .05. Fig. 6 contains the paths and their explanatory powers that are produced by the GAP algorithm (i.e., $\mathcal{PU}$ in Fig. 5). Note that it may also be desirable to report explanatory power by measurement variable. This is computed by summing for each measurement variable the explanatory power of all paths reported by the QPD algorithm that terminate with that variable.

Observe that the results displayed in Fig. 5 make it easy to generate explanations of performance problems in terms of the detection variable. For example, consider the path $RTTOT \rightarrow SJCPU \rightarrow STCPU \rightarrow$

$STCPU\_1$. Given this path and its u_P, it is relatively easy for a decision support tool to generate text that explains how $STCPU\_1$ contributes to increased response times. For example,

> User 1 accounts for 22% of the increase in response time. The reason is as follows:
>
> 1. User 1 accounts for 90% of the increase in CPU service time.
> 2. CPU service time accounts for 80% of the increased time in the CPU subsystem.
> 3. Time in the CPU subsystem accounts for 30% of the increase in response time.

To summarize, the GAP algorithm provides a systematic approach to quantitative performance diagnosis. In particular, the algorithm applies to very general MNGs (e.g., directed acyclic graphs, not just trees), and quantifies explanations in a way that can be interpreted as fractional contributions to the problems present.

3 Diagnostic Techniques

Diagnostic techniques quantify the effect on a parent variable of performance problems in its children. This section describes the GAP framework for constructing diagnostic techniques and analyzes some specific techniques.

To motivate the following discussion, consider $RTTOT = SJCPU + SJIO + SJPAGE + RTOTHER$ in the running example. Suppose that initially $RTTOT$ is 1.06 seconds. Subsequently, an administrative task begins execution and $RTTOT$ grows to 1.52 seconds. Diagnostic techniques provide a way to compute $w_{i,j}$, the weight of the arc in the MNG from x_i to x_j. In the context of this example, the arc weights should be interpreted as the fraction of the performance problem evidenced in $RTTOT$ that can be attributed to its children $SJCPU$, $SJIO$, $SJPAGE$, and $RTOTHER$. That is, $w_{0,1}$ is the fraction of the problem that can be attributed to $SJCPU$, and so on.

Several definitions are introduced to facilitate the following discussion. The term **target data** refers to measurements collected when a performance problem may be present. There are M observations of each variable in the target data, and $x_i(t)$ denotes the value of x_i at time t. For example, in Fig. 1 the level shift during 10:45-11:30 is a performance problem and hence a set of target data. There may also be

reference data that are collected when no performance problem is present. Let $x_i^r(t)$ be the value of the t-th observation of x_i in the reference data, where $1 \leq t \leq M^r$. In the running example, the reference data are the intervals between 1:00 and 2:00.

How do performance problems influence a measurement variable? To address this question, note that target data can be separated into two parts. The first is the **base value**, what the target value of the measurement variable would have been if there were no performance problem. The second is the **problem value**, the perturbation of the variable in the target data as a result of the performance problem. Let $y_k(t)$ denote the base value of the k-th measurement variable, and let $z_k(t)$ be the problem value of x_k. Thus, by definition

$$x_k(t) = y_k(t) + z_k(t). \tag{2}$$

In practice, it is rare to have measurements of either $y_k(t)$ or $z_k(t)$ and thus at least one of these must be estimated.

Typically, analysis of large data sets is done by first aggregating the data. The choice of the statistic that aggregates $x_k(1), \cdots, x_k(M)$ is denoted by $stat_{x_k}$. Examples of such statistics are the mean, variance, standard deviation, and distribution quantiles.

The approach herein taken to quantify the parent-child effects of performance problems is to estimate statistics of **problem values** based on observations contained in the target and reference data. Let $stat_{z_k}$ denote a statistic of z_k. (It is assumed that only finite valued statistics are used.) A diagnostic technique can be constructed by specifying the following:

1. a statistic (e.g., mean, variance) that aggregates problem values
2. the estimator of this statistic

Actually, for the above two considerations to be sufficient to specify a diagnostic technique, something else must be known—g_i, the functional relationship between the parent x_i and its children. While in some cases this is known (e.g., the relationship between $RTTOT$ and its children as well as between $TRIO$ and its children), in many cases it is not. The most widely used approach to handling such situations in practice is to approximate g_i by a linear function. Specifically, suppose that x_i has children $x_1, \cdots, x_n$, and so

$x_i = g_i(x_1, \cdots, x_n)$. The linearity assumption implies that

$$g_i(x_1, \cdots, x_n) = \sum_{j=1}^{n} (a'_{i,j}) x_j, \tag{3}$$

for constants $a'_{i,j}$. Such a linear approximation can work well even for non-linear relationships. For example, at low utilizations, sojourn times are a linear function of service times with $a'_{i,j} \approx 1/(1-\rho)$ (where ρ is the utilization of the server in the queueing system).

The linearity assumption deserves more discussion since queueing systems are highly non-linear. First, note that it is not required that linearity always hold. Rather, it is only required that linearity is a reasonable approximation for the values in the reference and target data. Further, smooth curves (such as response time curves) can be approximated by linear functions. Even so, it may be that the linearity assumption is unreasonable. This can be addressed in several ways. If the functional relationship between parent and child variables (i.e., g_i) is known, then a Taylor series approximation can be used as in [19]. If g_i's functional form is known but it contains unknown constants (e.g., visit rates), it is often possible to estimate these constants using least-squares regression. This is useful, for example, in transaction processing applications where routing between data servers is a function of the transaction being executed rather than the data in the form being processed. In other cases where g_i's functional form is unknown, it may be possible to approximate g_i. For example, capacity planning software often uses M/M/1 equations to approximate response times in a more complex queueing system (e.g., [2]).

At the heart of constructing diagnostic techniques is the ability to apportion problems evident in the parent variable to its children, especially for statistical aggregations. It turns out that if Eq. (3) is a reasonable approximation, then the following holds for many statistics of interest

$$stat_{z_i} = \sum_{x_j \in chld(x_i)} (a_{i,j}) stat_{z_j}. \tag{4}$$

For example, if *stat* is the mean, then $a_{i,j} = a'_{i,j}$. If *stat* is the variance and child variables as well as the $y_k(t), z_k(t)$ are mutually independent, then $a_{i,j} = a'^2_{i,j}$. Generalizations to the log-likelihood ratio (e.g., for hypothesis testing) and to percentiles are possible as well.

As indicated before, $stat_{z_k}$ is not observed directly and so it must be estimated. Let $\hat{stat}_{z_k}$ denote this

estimator. Thus, arc weights are computed as

$$\delta_{i,j} = \psi\left(sign(\hat{stat}_{z_i})(a_{i,j})\hat{stat}_{z_j}\right) \tag{5}$$

and

$$w_{i,j} = \begin{cases} \frac{\delta_{i,j}}{\sum_{x_k \in chld(x_i)} \delta_{i,k}} & \text{if } \sum_k \delta_{i,k} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where

$$\psi(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The term $\delta_{i,j}$ is included so that the effect of a child variable on the parent only includes those parts that relate to how the performance problem impacts performance.

To illustrate the foregoing, consider the example of $RTTOT$ with the statistic being the mean. Here, $a'_{i,j} = 1 = a_{i,j}$. Suppose that $z_{RTTOT} = 0.46$, $\hat{stat}_{z_{SJCPU}} = -0.12$, $\hat{stat}_{z_{SJIO}} = 0.50$, and $\hat{stat}_{z_{SJPAGE}} = 0.08$. Then, $\delta_{RTTOT,SJCPU} = 0$ ($SJCPU$ changes in a way that does *not* explain the performance problem), $\delta_{RTTOT,SJIO} = 0.50/0.58 = 0.86$, and $\delta_{RTTOT,SJPAGE} = 0.8/0.58 = 0.14$. More generally, If $\hat{stat}_{z_k} = \hat{\mu}_{z_k}$ (the estimate of the mean), then $\delta_{i,j} = \psi(sign(\hat{\mu}_{z_i})a'_{i,j}\hat{\mu}_{z_j})$. If $\hat{stat}_{z_k} = \hat{\sigma^2}_{z_k}$ (the population variance) and covariances are zero, then $\delta_{i,j} = a'^2_{i,j}\hat{\sigma^2}_{z_j}$.

Now consider how to obtain $\hat{stat}_{z_k}$. Two situations are of particular interest. In the first, the performance problem dominates the value of measurement variables so that

$$stat_{z_k} \approx stat_{x_k}. \tag{7}$$

Hence, it is reasonable to use $\hat{stat}_{z_k} = \hat{stat}_{x_k}$, where $\hat{\mu}_k = \bar{x}_k$ and $\hat{\sigma}^2_{x_k} = s^2_{x_k} = \sum_{t=1}^{M}(x_k(t) - \bar{x}_k)^2/(M-1)$. Thus, if the statistic is the mean,

$$\delta_{i,j} = \psi\left(sign(\bar{x}_i)a'_{i,j}(\bar{x}_j)\right). \tag{8}$$

If the statistic is the variance and covariances are zero,

$$\delta_{i,j} = a'^2_{i,j}s^2_{x_j}. \tag{9}$$

A second situation for computing $\hat{stat}_{z_k}$ is when there are reference data that are representative of the

Diagnostic Technique		*Arc Weight ($w_{RTTOT,j}$)*			
Statistic	*Estimator*	*SJCPU*	*SJIO*	*SJPAGE*	*RTOTHER*
μ_{z_i}	$\bar{x}_i$	.14	.23	.10	.53
μ_{z_i}	$\bar{x}_i - \bar{x}_i^r$	.13	.33	.17	.37
$\sigma^2_{z_i}$	$s^2_{x_i}$	.14	.12	.17	.50
$\sigma^2_{z_i}$	$s^2_{x_i} - s^2_{x_i^r}$	.18	.12	.13	.57

Figure 7: Arc Weights for Techniques In Running Example

base values of the measurement variables. That is, $stat_{y_k} \approx stat_{x_k^r}$. Here,

$$\hat{stat}_{z_k} \approx \hat{stat}_{x_k} - \hat{stat}_{x_k^r}. \tag{10}$$

So, if the statistic is the mean, $\delta_{i,j} = \psi\left(sign(\bar{x}_i - \bar{x}_i^r)a_{i,j}(\bar{x}_j - \bar{x}_j^r)\right)$ and for the variance (with small covariances), $\delta_{i,j} = \psi\left(sign(s^2_{x_i} - s^2_{x_i^r})a_{i,j}(s^2_{x_j} - s^2_{x_j^r})\right)$.

In sum, diagnostic techniques are determined by (a) the statistic used to aggregate data and (b) the manner in which problem values are estimated. Different choices for (a) and (b) result in different diagnostic techniques.

To illustrate the foregoing, consider the decomposition of $RTTOT$ into $SJCPU, SJIO, SJPAGE, RTOTHER$ in the running example with problem data are taken from Fig. 1 (10:45-11:30) and reference data from 1:00-2:00. This problem resulted from a software error related to managing one memory pool in the paging subsystem. As such, both the paging and CPU subsystems are affected.

To proceed, two statistics are considered, the mean and variance. Problem values are estimated using both the dominant value approach (i.e., $\hat{stat}_{z_k} = \hat{stat}_{x_k}$) and using reference data. With two possibilities for each component of the framework, there are four diagnostic techniques. The results are displayed in Fig. 7. Unfortunately, all of these techniques produce arc weights that are inconsistent with $SJCPU$ and $SJPAGE$ explaining the performance problem.

Using the GAP framework, a new diagnostic technique can be constructed. The intuition is that the techniques used above do a poor job of estimating problem values. This shortcoming might be overcome

if a statistic is used that incorporates the relationship between the parent and child variable, such as the covariance (cv). The proposed diagnostic technique uses covariance as the statistic and the difference between target and reference covariances to estimate this statistic for problem values. The resulting arc weights from $RTTOT$ are: 0.44 for $SJCPU$, 0.23 for $SJIO$, 0.33 for $SJPAGE$, and 0 for $RTOTHER$. This is a much better fit to what is expected for this performance problem.

4 Discussion and Extensions

The GAP algorithm described in Section 2 has several desirable properties. First, the algorithm is guaranteed to terminate if the MNG is acyclic (a property that is easy to verify). Second, if arc weights are normalized, then all paths have an explanatory power in $[0, 1]$ (which follows from the definition of a normalized path and the manner in which explanatory power is computed). Last, let $\mathcal{P}_0$ be the set of all paths from x_0 to the leaves of the MNG. It turns out sum of the explanatory power of these paths is 1. That is, $\sum_{P \in \mathcal{P}_0} u_P = 1$. This is straightforward to establish for a tree-structured MNG. For a directed acyclic graph, a bit more care is required.

Let $\mathcal{P}_{i,l}$ be the set of all paths from x_i to x_l and let V_L be the set of leaves (those measurement variables that have no child). It suffices to show the following holds for all $x_i \in V$:

$$\sum_{x_l \in V_L} \sum_{P \in \mathcal{P}_{i,l}} u_P = 1$$

The proof uses induction on the maximum distance of the path from x_i to a leaf. Let n denote this distance.

1. $n = 1$. Thus, $chld(x_i) \subseteq V_L$. The conclusion follows from the definition of normalized weights.

2. n implies $n + 1$. Observe that

$$\mathcal{P}_{i,l} = \bigcup_{x_j \in chld(x_i)} \left(\mathcal{P}_{j,l} \bigcup \{(x_i, x_j)\} \right)$$

Thus,

$$\begin{array}{rcl} \sum_{x_l \in V_L} \sum_{P \in \mathcal{P}_{i,l}} u_P & = & \sum_{x_j \in chld(x_i)} w_{i,j} \sum_{x_l \in V_L} \sum_{P \in \mathcal{P}_{j,l}} u_P \\ & = & \sum_{x_j \in chld(x_i)} w_{i,j} \\ & = & 1 \end{array}$$

(The second step follows from the induction hypothesis; the third step follows from the weights being normalized.)

Returning to the GAP framework for diagnostic techniques, it is useful to show that the framework provides a means to characterize existing approaches to QPD, especially what's-biggest analysis, what's-different analysis, and correlation analysis. What's-biggest analysis (e.g., see [7] for performance tuning [13] and for automating assistance with linear programming models) computes the contribution of child variables to performance problems as $a_{i,j}\bar{x}_j/\bar{x}_i$. In terms of the QPD framework, $stat_{z_k} = \mu_{z_k}$ and $\hat{stat}_{z_k} = \hat{stat}_{x_k}$. What's-different analysis (e.g., see [14] for performance tuning and [19] for corporate financial statements) quantifies the impact of performance problems by examining how *changes* in the mean of child variables affect their parents. Thus, $stat_{z_k} = \mu_{z_k}$ and $\hat{stat}_{z_k} = \hat{stat}_{x_k} - \hat{stat}_{x_k^r}$.

Correlation analysis (e.g., see [20] in the NaviGraph performance product and [15] for the Performance Analysis Facility/VM) estimates the strength of parent-child relationships based on the magnitude of the cross correlation between parent and child. That is, $w_{i,j} = \mid \frac{s_{x_i,x_j}}{s_{x_i} s_{x_j}} \mid$, where s_{x_i,x_j} is the sample covariance between x_i and x_j. Unfortunately, this is not a normalized weight since $\sum_j a_{i,j}\sigma_{x_j} \neq \sigma_{x_i}$. However, if covariances between child variables are small and there is a linear relationship between parent and child variables, then $\sigma^2_{x_i} \approx \sum_j a_{i,j}\sigma^2_{x_j}$. Thus, a normalized weight can be constructed using $w_{i,j} = \frac{a_{i,j}s^2_{x_j}}{s^2_{x_i}}$, where $stat_{z_k} = \sigma^2_{z_k}$ and $\hat{stat}_{z_k} = \hat{stat}_{x_k} = s^2_{x_k}$.

Last, it is insightful to develop a simple analytic framework in which diagnostic techniques can be compared. The focus of these comparisons is the choice of statistic for aggregating problem values. So, to simplify matters, $\hat{stat}_{z_k} = \hat{stat}_{x_k}$. As a further simplification, the parent variable x_0 has two children, x_1 and x_2 such that $x_0 = x_1 + x_2$. x_1 is dominated by a performance problem, and x_2 has no performance problem. Thus, it is desired that diagnostic techniques compute arc weights such that $w_{0,1} > w_{0,2}$. Put differently, the **effectiveness** of a diagnostic technique is $\omega = w_{0,1}/w_{0,2}$. Clearly, a larger value of ω is desired.

This simple framework provides for some interesting insights into the choice of the statistic for aggregating problem values. Suppose the statistic is the mean. Thus, from Eq. 6, $w_{0,1} = \frac{\bar{x}_1}{\bar{x}_1+\bar{x}_2}$ and $w_{0,2} = \frac{\bar{x}_2}{\bar{x}_1+\bar{x}_2}$. Hence, $\omega_\mu = \frac{\bar{x}_1}{\bar{x}_2}$. If the standard deviation is used (so that the units are the same units as the mean), then $\omega_\sigma = \frac{s_{x_1}}{s_{x_2}}$.

From this, it is easy to quantify the condition under which the diagnostic technique that uses the standard deviation has a larger effectiveness than the technique using the mean.

$$\omega_\sigma > \omega_\mu \text{ iff } cv_{x_2} < cv_{x_1}, \tag{11}$$

where $cv_x = s_x/\bar{x}$ is the coefficient of variation for $x(t)$.

Now consider how the choice of statistic is affected in the context of a specific performance problem. In particular, consider a **uniform magnitude problem profile (UMPP)** (e.g., the level shift in Fig. 1). In the situation being analyzed, $z_1(t) \in \{0, h_1\}$. Let f be the fraction of the intervals in which $z_1(t) = h_1$. Then,

$$\begin{array}{rcl} \bar{z}_1 & = & fh_1 \\ s^2_{z_1} & = & h_1^2 f(1-f) \\ cv_{z_1} & = & \sqrt{\frac{1-f}{f}}. \end{array} \tag{12}$$

From Eq. (11),

$$\omega_\sigma > \omega_\mu \text{ iff } cv_{x_2} < \sqrt{(1-f)/f}.$$

Thus, for UMPPs, if the target data only includes times when there is a performance problem (i.e., $f \approx 1$), then diagnostic techniques based on the mean are more effective than those that use the variance. On the other hand, if only a few intervals contain performance problems (i.e., f is small), diagnostic techniques based on the variance are more effective.

5 Conclusions

This paper describes GAP, a general approach to quantitative performance diagnosis. GAP has two parts: (1) an algorithm for computing quantitative performance diagnoses and (2) a framework for constructing diagnostic techniques that provides the basis for quantifications produced by the algorithm.

The GAP algorithm is based on the concept of a measurement navigation graph (MNG), a directed acyclic graph whose nodes are measurement variables and whose arcs have weights that quantify the effect of child variables (e.g., LAN utilization) on parent variables (e.g., response time). The algorithm allows for non-tree structured graphs, such as those that arise if there are shared resources. Various properties of the

algorithm are established, especially that its quantification of explanations can be interpreted as fractional contributions to the performance problem.

Arc weights in the MNG are computed by diagnostic techniques. A framework for developing diagnostic techniques is described that consists of (a) the choice of statistic (e.g., mean, variance) to aggregate problem values and (b) the estimator of this statistic. The framework is used to analyze existing diagnostic techniques. One result is that correlation analysis does not produce normalized weights and so cannot be used to construct fractional explanations. A modified version of correlation analysis is introduced that addresses this shortcoming, and it is shown to produce superior results in diagnosing a performance problem in a production computing system. Also presented is an analysis of the choice of statistic for diagnostic techniques that provides insight into the trade-off between using the mean and the standard deviation in the specification of a diagnostic technique. Specifically, it is shown that for uniform magnitude performance problems (e.g., a step), the standard deviation is preferred to the mean if the problem data have a coefficient of variation no larger than $\sqrt{(1-f)/f}$, where f is the fraction of the data containing the performance problem.

The paper raises a number of issues that are being explored. Some issues relate to assumptions made in the approach to constructing diagnostic techniques, such as: (a) the adequacy of linear models between parent and child variables, (b) the magnitude of covariances between child variables, and (c) the covariance structure of base and problem values. These issues will be addressed by obtaining data from operation systems, both when performance problems are present and when they are not. More broadly, follow-on work is being pursued related to: (i) additional diagnostic techniques, (ii) the construction and evolution of MNGs (e.g., when adapting to changes in systems and measurement sources), (iii) diagnosis for systems when there are multiple detection variables, (iv) diagnosing problems that affect the structure of the MNG, such as a new software release that changes where queueing takes place, and (v) application to systems where information is decentralized.

Acknowledgements

We thank Robert F. Berry and Norbert Lenz for their thoughtful comments on earlier drafts of this paper.

References

[1] **M.F. Abu El-Yazeed**: *Minimum Measurements at Minimum Set of Test Nodes for Analog Circuit Fault Diagnosis*, 11th IEEE Mediterranean Electorical Conference, Vol 1., 510-516, 2002.

[2] **Subhash Agrawal**: *Metamodeling: a Study of Approximations in Queueing Models*, MIT Press, 1985.

[3] **B. Arinze, M. Igbaria, and L.F. Young**: "A Knowledge Based Decision Support System for Computer Performance Management," *Decision Support Systems* **8**, 501-515, 1992.

[4] **O. Benkahla, C. Aktouf, and C. Robach**: "Performance Evaluation of Distributed Diagnosis Algorithms in Parallel Systems," *Paralllel Computing*, Vol. 24, No. 8, 1205-22, 1998.

[5] **Robert F. Berry and Joseph L. Hellerstein**: "A Unified Approach to Interpreting Measurement Data in Performance Management Applications," *First IEEE Conference on Systems Management*, University of California, Los Angeles, May, 1993.

[6] **Robert F. Berry and Joseph L. Hellerstein**: "An Flexible and Scalable Approach to Navigating Measurement Data in Performance Management Applications," *Proceedings of the Second IEEE International Conference on Systems Management*, June, 1996.

[7] **Boole & Babbage**: *DASD ADVISOR Technical Backgrounder*, Boole & Babbage, Inc., 10BAB34.BGR, December, 1987.

[8] **Tom Bowen and Liz Payling**: "Expert Systems for Performance Review," *Journal of Operations Research Society*, **38** 929-934, 1987.

[9] **Computer Associates**: "CA-MINDOVER," Computer Associates, 711 Stewart Avenue, Garden City, New York, 1993.

[10] **Johan De Kleer, Alan K. Mackworth, Raymond Reiter**: "Characterizing diagnoses and systems," *Artificial Intelligence*, **56** 197-222, 1992.

[11] **Bernard Domanski**: "A PROLOG-based Expert System for Tuning MVS/XA," *Proceedings of the Computer Measurement Group*, 160-166, 1987.

[12] **The Gartner Group**: "Mainframes, Minis, and Micros," The Gartner Group, Stamford CT., 1993.

[13] **Harvey J. Greenberg**: "Rule-based Intelligence to Support Linear Program Analysis," *Decision Support Systems*, **9**, 425-447, 1993.

[14] **J.L. Hellerstein**: "What's-Different Analysis and Its Application to Performance Management," *Proceedings of the Computer Measurement Group*, 1988.

[15] **IBM**: *VMPAF User's Guide and Reference Manual*, (SC23-0564-00) IBM Corporation, 1993.

[16] **D. R. Irwin**: "Monitoring the Performance of Commercial T1-rate Transmission Service," *IBM Journal of Research and Development*, 805-814, 1991.

[17] **R. Isermann and B. Freyermuth**: "Process Fault Diagnosis Based on Process Model Knowledge," *Proceedings of 1989 ASME International Computers In Engineering Conference and Exposition*, July, 631-642, 1989.

[18] **R.A. Johnson and D.W. Wichern**: *Applied Multivariate Statistical Analysis*, Prentice Hall, 1988.

[19] **Donald W. Kosy and Ben P. Wise**: "Self-Explanatory Financial Planning Models," *Proceedings of the National Conference on Artificial Intelligence*, 176-181, 1984.

[20] **Landmark Systems Corporation**: "NaviGraph," Landmark Systems Corporation, 8000 Towers Crescent Drive, Vienna, Virginia, 1993.

[21] **Robert Milne**: "Using AI in the Testing of Printed Circuit Boards," *National Aerospace and Electronics Conference*, Dayton, Ohio, May, 1984.

[22] **I. V. Nikiforov**: "On-line Diagnosis of Dynamic-Stochastic Systems," *IFAC Fault Detection, Supervision and Safety for Technical Processes*, Espoo, Finland, 299-304, 1994.

[23] **J. Olsson**: "An Architecture for Diagnostic Reasoning Based on Causal Models," *The Royal Institute of Technology*, Department of Computer and Systems Sciences, Report No. 91-004-DSV.

[24] **R. Reiter**: "A Theory of Diagnosis From First Principles," *Artificial Intelligence*, Vol. 32, No. 1, 57-95, 1987.

[25] **Bherokh Samadi**: "TUNEX: A Knowledge-Based System for Performance Tuning of the UNIX Operating System," *IEEE Transactions on Software Engineering*, Vol. 15, No. 7, 1989.

[26] **M. Sivaraman and A.J. Strojwas**: "Path Delay Fault Diagnosis and Coverage: A metric and an Estimation Technique," *IEEE Transactions on Computer Aided Design Integration and Circuit Systems*, Vol. 20, No. 3, 440-457, 2001.

[27] **Douglas Smith**: "Monitoring/diagnostic Systems Enhance Plant Asset Management," *Power Engineering*, 23-28, June 1992.

[28] **M. Stanek, M. Morari, and K. Frohlich**: "Model-Aided Diagnosis: An Inexpensive Combinaiton of Model-Based and Case-Based Condition Assessment," *IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews* Vol. 31, No. 2, 137-145, 2001.

[29] **B. W. Surgenor, JI Glasgow, M. Feret**: "Methodologies for Generic Fault Diagnosis," *17th International Conference on Applications of Artificial Intelligence in Engineering*, 609-625, 1992.

[30] **Walter Vandaele**: *Applied Time Series and Box-Jenkins Models*, Academic Press, Inc., 1983.

[31] **Jie Ying, T. Kirubarajan, Krishna R. Pattipati, and An Patterson-Hine**: "A Hidden Markov Model-Based Algorithm for Fault Diagnosis with Partial and Imperfect Tests," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 30, No. 4, 463-473, 2000.