# Fault Isolation during Semiconductor Manufacturing using Automated Discovery from Wafer Tracking Databases *

Sharad Saxena
Texas Instruments
P. O. Box 655012, MS 461, Dallas, TX 75265

## Abstract

*This paper describes the use of automated discovery from databases for diagnosing the causes of misprocessing during semiconductor manufacturing. The database contains the history of the semiconductor wafers as they undergo various processing steps. A generate-and-test approach is taken for using such a database for automated diagnosis. Based on prior manual use of such databases, classes of queries to the database useful for fault isolation are identified. Patterns in the responses to these queries that are useful for fault isolation are also identified. Automated diagnosis is accomplished by automating query generation and the detection of potentially useful patterns. A prototype system was implemented and tested on a database from a wafer grinding and polishing facility. In addition to identifying previously known faults, the system also identified previously unknown faults.*

Topic: Data Discovery & Database Mining, Diagnosis.
Domain: Semiconductor Manufacturing.
Language/Tool: Smalltalk, Unix, Sun workstation.
Status: Prototype developed and tested. Larger implementation in progress.
Effort: Prototype developed in 2 man-months.
Impact: Has the potential for assisting in problem diagnosis during semiconductor manufacturing.

## 1 Introduction

One goal of automated discovery from databases is to automatically sift through a large database and bring "interesting" and/or "useful" facts to the attention of a person. Since interestingness and utility are domain-dependent and subjective concepts, a general treatment of the issues related to automated discovery from databases is difficult. The task is simplified by restricting attention to a specific domain and identifying a set of queries, and the patterns among corresponding responses, that would be considered interesting in the domain. An automated system can then be constructed for sifting through the database to identify the presence of these patterns among the responses to the chosen queries. In addition, experience in identifying interesting patterns in a number of specific domains may provide clues about a general theory of interestingness. Such a theory can then be applied to develop more powerful methods for automated discovery in databases. Motivated by these issues we have developed a prototype system that applies the above approach to the task of fault diagnosis during semiconductor manufacturing. This paper describes the design and implementation of this prototype system.

Semiconductor manufacturing is a long, expensive, and complicated process. State of the art integrated circuits may require between 200-300 processing steps. Slight deviation from the desired performance of any one of these steps can produce unacceptable product. Since process deviations and equipment malfunctions are inevitable, rapid diagnosis of the causes of misprocessing becomes critical to the profitable manufacturing of semiconductors. However, the complex interrelationships between the various processing steps makes fault isolation difficult. Measurements made on a wafer at the end of a long sequence of steps are often insufficient for diagnosis.

The effects of different processing steps on the final product can be partially separated if one records in a database the history of the wafer as it undergoes different processing steps. This database associates the following information with each wafer: identifying numbers for different machines that process the wafer, the time at which the different processing steps are performed, the name of operators for each machine, exact processing conditions, and any other information that one thinks might be useful for problem identification. Such a database is called a *wafer-tracking* database.

Later, when say the product yields become unac-

ceptable, one can obtain the following information from a wafer tracking database:

- What is common to all wafers having a low yield?
- What is common to all wafers having a certain measurement?
- With respect to a certain measurement, do all the wafers processed by certain machines have values much different from the wafers processed by other machines?
- Is there a trend in a certain measurement as more wafers are processed by a particular machine?

Answers to the above questions, and other similar questions, can help in rapid identification and correction of the causes of misprocessing.

Unfortunately, to use a wafer tracking database for fault isolation a large number of queries of the kind illustrated above that must be generated and evaluated for their fault-isolation potential. Consequently, when performed by a person, the use of wafer tracking databases for diagnosis can become quite tedious, time-consuming, and error-prone. Section 2 of this paper describes our approach for automatically identifying diagnostic information from a wafer tracking database. Section 3 describes the design of a prototype system that implements our approach. The prototype was tested on a database containing three months' history of wafers in a grinding and polishing area of a crystal growth facility at Texas Instruments. In addition to identifying the previously known faults correctly, the automated system also pointed out defects that were previously unknown. Sections 4 presents these results. Section 5 concludes with a summary and some directions for extending this work.

## 2 Automated Discovery from Wafer Tracking Databases

The use of wafer tracking databases for diagnosing causes of misprocessing during semiconductor manufacturing was pioneered in the Wafer Sleuth system by Scher et al: [1, 2]. The Wafer Sleuth system employed optical character recognition to read an identifying number for each piece of processing equipment, and the location of the wafer within the equipment, as the wafer was processed by a particular piece of equipment. Later, this information was used to suggest causes of mis-processing. Scher et al. give numerous examples where wafer tracking information has been helpful in rapidly identifying problems in a wafer fabrication facility.

One limitation of the Wafer Sleuth system is that the task of making queries to the wafer tracking database is manual, as is the task of determining whether a query contains information helpful for fault isolation. Because of the large number of processing steps required to manufacture a modern integrated circuit, and the large number of parameters tested before accepting the product, the manual process of using wafer tracking databases can be quite tedious, time consuming, and error prone.

We take a generate-and-test approach to automate the task of identifying useful queries from a wafer tracking database. The task is broken into two main components; *query generation*, which generates the set of possible queries; and *query evaluation*, which evaluates the interestingness of each query. A query is considered interesting for diagnosis if the response to that query has fault isolation potential.

A query generator can be constructed by identifying the classes of queries that will be useful for fault isolation. Depending on the available computational resources (especially time), the query generator can exhaustively generate all queries in the previously identified classes, or may incorporate heuristics about queries more likely to to be interesting. The query generator can also base the query generation on the feedback obtained from previous queries considered interesting. The query generator can also have domain filters to prevent generation of queries that are known a-priori to be uninteresting. For example, for each query regarding a particular fault one can count the total loss of product yield due to that fault. Any query for which the total yield loss is below a certain threshold can be removed from the process of query evaluation.

The query evaluator determines whether a particular query is interesting. Recall that a query is considered interesting if its response has the fault-isolation potential. Based on the prior manual use of Wafer Sleuth, one can identify two broad classes of query responses that have been useful for fault isolation in the past. The first class of interesting responses are those in which a fraction of the machines, and/or wafers, from a set behave differently from the rest. For instance, say a fabrication facility has five ion implanters A, B, C, D, and E. Suppose that the average threshold voltage of the transistors formed using these implanters is as shown in Figure 1. From this figure one concludes that implanter B is producing transistors with much lower threshold voltage than the other implanters, suggesting that implanter B needs repair. Stated differently, Figure 1 is interesting because one

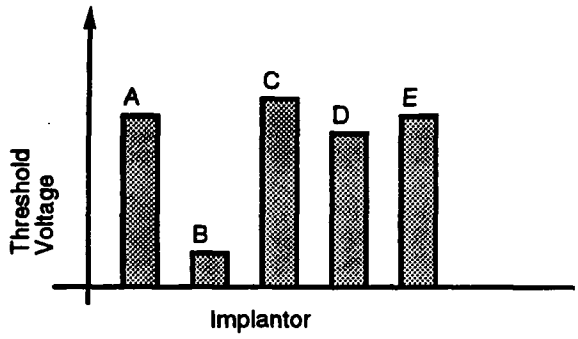


Figure 1: Fault isolation due to outliers

data point is different from the others. Generalizing this observation, one concludes that query responses containing a small number of *outliers*, or multiple *clusters*, have the potential for fault-isolation. Such queries are therefore considered interesting.

The second class of interesting responses are those where there is a *trend* in the value a certain parameters. This class of queries is useful when the successive measurements can be ordered. For the classes of queries considered in the prototype an ordering of query responses was not meaningful. However, the need for a trend detector can be anticipated for classes of queries more general than those considered in the prototype. Section 3 gives an example of how a query of this type helps in fault isolation.

The above two interestingness detectors are not exhaustive. These two are the ones that have been implemented in the prototype. New classes of interestingness detectors can be easily added to the library implemented in the prototype. However, we have found that in most of the cases known to us where wafer tracking has been successful, the response to a fault isolating query belongs to one of the above two cases. The approach of implementing detectors for known patterns is similar to the approach of having specific knowledge generation operators, as suggested by Kaufmann et al. [3]; and the approach of looking for specific statistical properties of data contained in a database, as proposed by Paitetsky-Shapiro and Matheus [4].

Based on some domain characteristics often one might want to prune the set of queries considered interesting by the purely data-driven interestingness detectors. For example, if the interestingness detectors can rank the various queries, one might like to see only those queries that score greater than a specified threshold on the interestingness score. *Domain specific filters* are provided for additional pruning of queries considered interesting by the interestingness detectors. Figure 2 illustrates the architecture incorporating the different components of the generate-and-test strategy used for diagnosis from wafer tracking databases. The next section describes a prototype that implements parts of this architecture.

## 3 Implementation of a Prototype

A prototype was built and tested using a database containing wafer tracking information gathered for three months in a wafer grinding and polishing facility at Texas Instruments. Except for the query generator, the rest of the prototype is general and does not depend upon the particular database.

Figure 3 shows the steps performed to produce wafers from semiconductor grade single crystal silicon. Typically, the grinding and polishing operations are done on a number of machines and each machine may have a number of heads. At the Texas Instruments facility the following information is recorded for each wafer as it goes through the grinding and polishing process: serial number, polisher number, head in the polisher, time of day, chemical properties of the slurry, etc. In addition, a wafer is inspected at a number of inspect stations. The defect code, if any, for each wafer is recorded at each station. All this information together constitutes the database for this very small part of the overall process of wafer fabrication. The next few subsections give the detail of the various sub-components of the architecture as they were implemented in the prototype.

### 3.1 Query Generator

In the previous manual use of wafer tracking databases at Texas Instruments, tabulation of different faults for each machine and sub-assembly within the machine had facilitated fault isolation. In the prototype the query generator exhaustively generated all possible queries of this type. For each inspect station and fault code a query was generated asking for the number of wafers processed by different heads of the different polishers having that fault. For instance, a typical query would be: "Give the number of wafers polished by the head number 3 on polisher A that had defect code 8 during the final inspect." Assuming that all polishers are uniformly loaded during normal operations, a subset of heads producing a large number of defective wafers suggests that these heads need maintenance.

### 3.2 Domain Filters

For each query corresponding to a particular defect, the total number of wafers having that defect

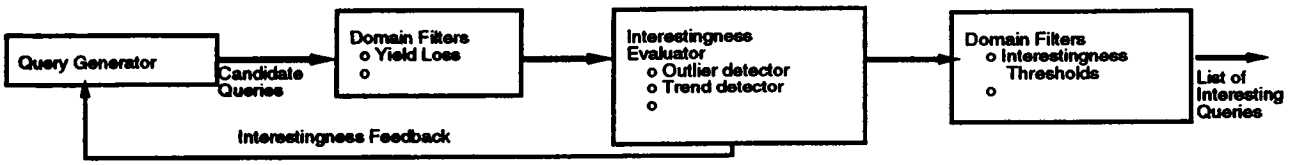


Figure 2: The architecture for automated diagnosis from Wafer Tracking databases

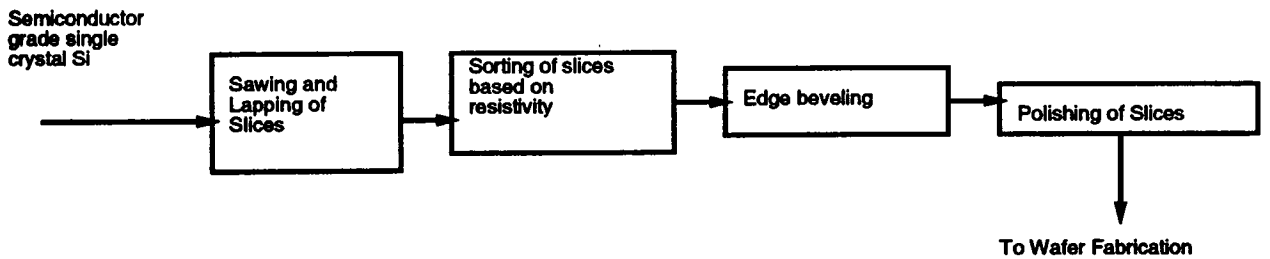


Figure 3: Process involved in making wafers from a crystal

was calculated. All queries for which the total number of wafers having the given defect less than a certain threshold were removed. For instance, since the total number of wafers having defect code 15 during the final inspect was less than a threshold, this query was not tested for interestingness. This is an example of a yield loss filter. Another domain filter considers only those queries containing outliers to be interesting that point towards pieces of equipment producing more faults that the rest. Query responses that show that some pieces of equipment are producing fewer faults than the rest are not considered interesting.

### 3.3 Interestingness Detectors

If a query is not filtered by the domain filters, it is tested for interestingness. Recall that a query is considered interesting if its response has fault-isolation potential. The result of the query generation process is a list of numbers. Each number represents the number of defective wafers of a particular type produced by the corresponding head and polisher. Two interestingness detectors were implemented; outlier and cluster detectors, and trend detectors.

#### 3.3.1 Outlier and Cluster Detector

The goal of outlier and cluster detection is to determine whether the response to a query can be partitioned into multiple clusters, or whether the response contains any outliers. There is a fairly extensive literature on clustering and outlier detection [5, 6]. Any of these algorithms that meet the the special requirements of our domain could be used. The special requirements of our domain are that one cannot assume any particular probability distribution for doing the cluster analysis, and one does not know the number of clusters a-priori.

Our approach to the clustering/outlier detection task is based on an application of the *minimum description length* principle. Recent investigations in artificial intelligence, theoretical computer science, and mathematical statistics have shown that under fairly general conditions, for a given language, among all models that can be described in the language, the model that enables the observations to be described most compactly makes the most accurate predictions [7, 8, 9, 10].

In our domain, we consider a query interesting if the response contains some values different from the majority of the values in the response. To determine the majority value, one first converts the measurements expressed as real numbers into integers by multiplying by a fixed precision. For instance, if at most two decimal digits of precision are considered significant then the numbers are multiplied by 100. Next, each possible integer between the maximum and minimum is considered as a potential *candidate* for the majority value. A language for describing integers is chosen such that the description length of $x$ when $\hat{x}$ is the candidate is a non-decreasing function of $x - \hat{x}$. Intuitively, this language expresses the property that one expects the likelihood of observing $x$ to decrease the further one gets from the candidate for majority $\hat{x}$. Elias's [11] variable length code for integers is one such language. The candidate that minimizes the number of bits required to describe all the observations with

Table 1: EliasCodelength for the positive integer $j$

```
length ← 1
while (⌊log j⌋ ≠ 0) {
length ← length +(1 + ⌊log j⌋);
j ← ⌊log j⌋ }
return(length)
```

this language is taken to be the estimate of the majority value.

We have found that the above procedure gives robust estimates of *mode* even in the presence of outliers. Table 1 gives the algorithm for determining the length of the Elias code for a positive integer. Given this procedure, the estimate of the mode of the observations $(X_1, X_2, \ldots, X_n)$ can be expressed mathematically as:

$$\hat{\mu} = \arg\min_{\mu} \sum_{i=1}^{n} \text{EliasCodelength} \mid (X_i - \mu) \mid .$$

where $\arg\min_{\mu}$ of an expression denotes the argument $\mu$ that minimizes the expression.

Once the robust estimate of the mode has been obtained, one estimates the deviation of the majority of the values in a similar manner. The deviation $d_i$ of the observation $X_i$ from $\hat{\mu}$ is measured as:

$$d_i = \text{EliasCodelength}(\mid X_i - \hat{\mu} \mid).$$

This results in the a set of deviations, $d_1, d_2, \ldots, d_n$. The mode of these deviations is the deviation of the majority of the values. The mode is estimated using the procedure outlined in the previous paragraph. Let $\hat{\sigma}$ denotes this estimate of the mode. Observations $X_i$ that have deviation $d_i$ greater than $\hat{\sigma}$ are considered to be outliers.

In addition to identifying queries containing fault-isolation information, one would also like to rank the queries so that queries more likely to help in fault isolation are ranked higher than the queries considered less likely to help in fault isolation. Queries whose responses are considered to contain outliers are ranked based on the amount of "outlierness". If $Y_1, \ldots, Y_k$ are the points considered as being outliers, and $\hat{\mu}$ is the estimate of the mode, then the measure of outlierness is computed as:

$$\text{Interestingness Score} = \sum_{i=1}^{k} 2^{\left|\frac{Y_i}{1+\hat{\mu}}\right|}$$

The weight assigned by this function to an outlier increases as the absolute value of the ratio outlier from the mode increases. Queries whose responses score higher values on the above measure for outlierness

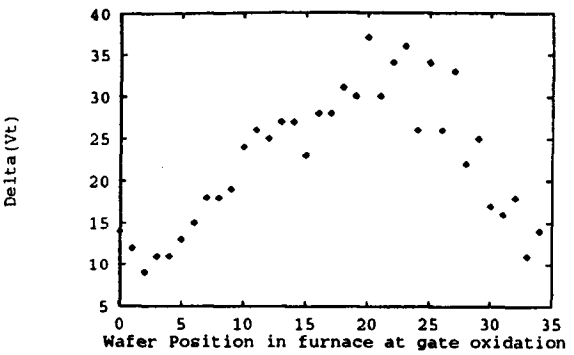


Figure 4: Example illustrating the need for considering subsequences in trend detection

are considered more interesting than queries whose responses score a lower value.

The procedures described above for identifying queries containing outliers and ranking them based on the outlierness were selected because they satisfied the requirements of our domain, and produced an acceptable ranking of the fault isolation potential of the different queries. It may be the case that other procedures for detecting outliers and outlierness ranking may perform equally well in this domain.

### 3.3.2 Trend Detector

Trend detection was not meaningful for the classes of queries considered in the prototype. It was included in the prototype because we think it will be useful for other classes of queries. Figure 4 shows an example taken from Scher [1] showing how trends in wafer tracking data can enable fault diagnosis. This figure shows the difference between threshold voltage at the top of the wafer and at the bottom of the wafer, plotted versus wafer position in the furnace at a key step in forming a transistor; namely gate oxidation. The threshold voltage at the bottom of the wafer was consistently lower than the top, and low threshold voltage results in an unacceptable device. Noticing that the trend peaked at the center enabled the engineers to locate the root cause of the problem.

Two complications arise in detecting whether there is a trend in a point cloud representing a query. First, the trend may not be linear. As a result, a linear correlation coefficient cannot be used for detecting a trend. This can be handled if one uses a non-parametric test for a trend. Kendall's $\tau$-coefficient is one such test [12, 13]. The value of the $\tau$-coefficient can be used as a numeric measure of the amount of trend. Queries

| p3 | (p1p2p3) | p1 (p2p3) | p1p2 (p3) |
|---|---|---|---|
| p2 | (p1p2)p3 | p1 (p2)p3 | |
| p1 | (p1)p2p3 | | |
| | p1 | p2 | p3 |

Table 2: Systematic consideration of subsequences to detect trends

with higher values of this coefficient are considered more interesting than the queries with a lower value.

The second problem is that a trend may exist only in a small segment of a point cloud. Collectively, the point cloud may not appear to have a trend based on Kendall's test. For instance, if one evaluates the complete point cloud shown in Figure 4 using Kendall's test the conclusion is that there is no trend at the 99% significance level. This is because the increasing and the decreasing trends are confounded in one statistic. This problem is handled by systematically considering various subsequences, and evaluating each subsequence for a trend. Table 2 summarizes this procedure with an example involving three points p1, p2, and p3. Only the points within () are checked for a trend. For example, in the entry in the column labeled p1 and row labeled p2, only the points p1 and p2 are checked for a trend. The interestingness score for the query is taken to be the score for the subsequence that has the highest value of the $\tau$-coefficient. With this algorithm the point cloud shown in Figure 4 is found to have a trend significant at the 99% level and it gets the score of 0.928 out of a maximum possible of 1. No effort is made to compare the interestingness scores of produced by different detectors. The goal is to produce a list of queries considered interesting by different types of detectors.

## 4 Results

The prototype produced a list of interesting queries sorted in the decreasing order of interestingness score from the database for each of the three months. This section reports the result obtained for one of these three months, similar results were obtained from the databases for the other two months. In the database for the month reported here, out of a set of 60 possible queries the system identified 14 queries to be interesting due to the presence of outliers. Trend detection was not attempted on this database because an ordering of the polishers and the heads within the polishers is not meaningful. Table 3 lists top 10 of these 14 queries along with their interestingness score.

Figure 5 graphically displays the query considered

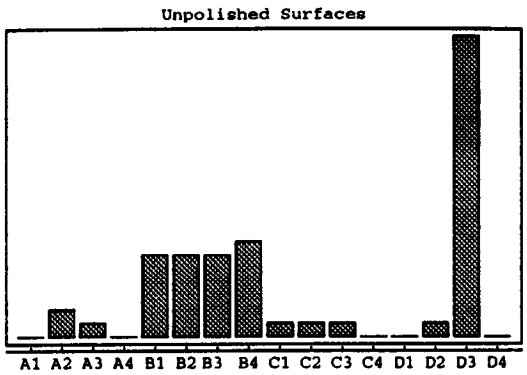


Figure 5: Query considered to be most interesting: Number of wafers with unpolished surface vs. head in a polisher. (Exact numerical values are not shown on the Y-axis to protect proprietary information.)

to be the most interesting due to the presence of outliers. For each head in a polisher, the plot shows the number of wafers processed by that head that were found at the final inspection to be unpolished. This figure shows that most of the unpolished wafers come from the third head in polisher D; suggesting that the head D3 needs repair. Furthermore, this figure also shows that, excluding the head D3, polisher B is producing more unpolished wafers than other polishers. This suggests that polisher B may need attention soon. A similar fault had also been discovered manually by the engineers at Texas Instruments. Figure 6 shows the plot that resulted in the manual discovery of the problem in head D3. Notice that this query is also on the list of automatically generated interesting queries shown in Table 3 (query number 8).

In addition to the above two queries, the prototype identified 12 other queries as being potentially helpful in fault isolation. Figure 7 shows one such query. This figure shows that among the wafers found to be thin at the sorting station a substantially large fraction come from head 4 of polisher A; suggesting that the head A4 needs repair. We had not been previously told about this problem, so, in a sense, this is a discovery by the system.

We have found the queries with low values of the interestingness score to be difficult to interpret for fault-isolation. This suggests that one could only consider those queries that have interestingness score above a certain number to be interesting. Alternatively, one could prune the list of interesting queries to show only the top few queries.

| Number | Inspect Station | Defect | Interestingness Score |
|---|---|---|---|
| 1 | Final | Unpolished Surface | 2083.31 |
| 2 | Final | Haze | 120.00 |
| 3 | Sort | Low Resistivity | 116.00 |
| 4 | Flat | Flatness | 93.88 |
| 5 | Flat | Warp | 59.96 |
| 6 | Sort | Other | 37.74 |
| 7 | Sort | Thin | 37.32 |
| 8 | Final | Dings | 31.71 |
| 9 | Sort | Back Contamination | 22.07 |
| 10 | Final | Polish Scratched | 15.29 |

Table 3: Top ten queries considered interesting due to outliers

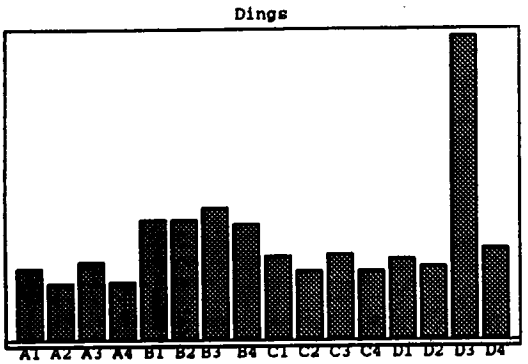


Figure 6: Manually discovered fault: Number of wafers with dings vs. head in a polisher. (Exact numerical values are not shown on the Y-axis to protect proprietary information.)

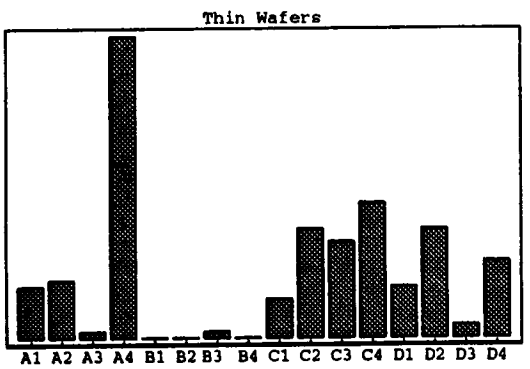


Figure 7: Number of thin wafers vs. head in a polisher. (Exact numerical values are not shown on the Y-axis to protect proprietary information.)

## 5 Conclusions

This paper described a prototype system that uses wafer tracking databases for diagnosing failures during semiconductor manufacturing. The prototype implements a generate-and-test approach for discovering interesting patterns from databases. In this approach one first determines a set of patterns that would be considered interesting in the domain, and then implements automatic procedures for detecting the presence of these patterns in the database. The prototype was tested on a database from a wafer grinding and polishing operation. In addition to identifying known faults, the prototype also identified faults that were previously unknown to us.

There are a number of directions in which this initial prototype can be extended. Developing more interestingness evaluators is an obvious direction. For numeric data coming from physical sources domain independent interestingness measures are also possible. Projection pursuit attempts this task by considering any non-Gaussian set of points to be interesting [14]. Other extensions would involve domain filters for pruning the queries before testing them for interestingness, and domain filters for pruning queries considered interesting by purely data-driven interestingness detectors. A major area that has not been addressed in the prototype is the issue of query generation. An exhaustive approach will be inadequate in complicated domains with a large number of parameters and machines.

In the domain of semiconductor manufacturing the benefits of automating the search for known patterns in a database are apparent. The task of fault isolation using wafer tracking databases is repetitive and some of the operations performed in this task can be specified precisely. By automating these repetitive and precisely specified tasks, the busy work required for

the effective use of wafer tracking database is reduced. This allows the process engineers to concentrate on more subtle faults. As a history of patterns in the responses to queries used to isolate these faults is accumulated, these new classes of queries and new interestingness evaluators can be added to the system, enabling one to improve the performance of such a system over time. Furthermore, a library of such patterns may provide us with a better understanding of the patterns considered interesting in this domain.

The widespread use of computers in today's workplace makes it easy to store and manipulate large quantities of data. However, due to the sheer volume of data, and the lack of appropriate tools for using data available in such large quantities, much of the recorded data is never utilised. We believe that a part of this problem can be alleviated by providing tools for automatically converting large quantities of data into useful information. We hope to have illustrated in this paper how one may go about doing this in the context of wafer tracking databases.

## Acknowledgments

This work was sponsored in part by the Air Force Wright Laboratory and DARPA Microelectronic Technology office under contract F33615-88-C-5448. The author thanks Nick Tovell, Amy Unruh, and Raj Wall for discussions and suggestions about the work reported here. The author also thanks Rick Cromer and Joe MacGrath for explaining the use of wafer tracking and providing the database.

## References

[1] G. Scher, D. H. Eaton, B. R. Fernelius, J. Sorenson, and J. W. Akers. In-line statistical process control and feedback for VlSI integrated circuit manufacturing. *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, 13(3):484–489, September 1990.

[2] G. M. Scher. Wafer tracking comes of age. *Semiconductor International*, May:126–131, 1991.

[3] K. A. Kaufman, R. S. Michalski, and L. Kerschberg. An architecture for integrating machine learning and discovery programs into an data analysis system. In G. Paitetsky-Shapiro, editor, *Proceedings of the 1991 AAAI workshop on Knowledge Discovery in Databases*, Anahiem, CA, 1991.

[4] G. Paitetsky-Shapiro and C. Matheus. Knowledge discovery workbench: An exploratory environment for discovery in business databases. In G. Paitetsky-Shapiro, editor, *Proceedings of the 1991 AAAI workshop on Knowledge Discovery in Databases*, Anahiem, CA, 1991.

[5] K. Fukanaga. *Introduction to Statistical Pattern Recognition.* Academic Press, San Diego, CA, second edition, 1990.

[6] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier detection.* John Wiley & Sons, New York, 1987.

[7] R. J. Solomonoff. A formal theory of inductive inference, part 1 and part 2. *Information and Control*, 7:1–22 and 224–254, 1964.

[8] M. Li and P. M. B Vitányi. Inductive reasoning and Kolmogorov complexity (preliminary version). In *Proceedings of the Fourth Annual IEEE Structure in Complexity Theory Conference*, pages 165–185, 1989.

[9] J. Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific, New Jersey, 1989.

[10] C. S. Wallace and P. R. Freeman. Estimation and inference by compact coding. *Journal of Royal Statistical Society, Series B*, 49:240–265, 1989.

[11] P. Elias. Universal codeword sets and representations of integers. *IEEE Transactions on Information Theory*, IT-24:194–203, 1971.

[12] J. D. Gibbons. *Nonparametric Statistical Inference.* MacGraw Hill, New York, 1971.

[13] M. G. Kendall. *Rank Correlation Methods.* Hafner Publishing Company, New York, 1962.

[14] P. J. Huber. Projection pursuit (with discussion). *The Annals of Statistics*, 13(2):435–525, 1985.