MDPI

Article

Chip-Level Defect Analysis with Virtual Bad Wafers Based on Huge Big Data Handling for Semiconductor Production

Jinsik Kim and Inwhee Joe *

Department of Computer Science, Hanyang University, Seoul 04763, Republic of Korea; x1m2n3o4p@hanyang.ac.kr
* Correspondence: iwjoe@hanyang.ac.kr

Abstract: Semiconductors continue to shrink in die size because of benefits like cost savings, lower power consumption, and improved performance. However, this reduction leads to more defects due to increased inter-cell interference. Among the various defect types, customer-found defects are the most costly. Thus, finding the root cause of customer-found defects has become crucial to the quality of semiconductors. Traditional methods involve analyzing the pathways of many low-yield wafers. Yet, because of the extremely limited number of customer-found defects, obtaining significant results is difficult. After the products are provided to customers, they undergo rigorous testing and selection, leading to a very low defect rate. However, since the timing of defect occurrence varies depending on the environment in which the product is used, the quantity of defective samples is often quite small. Unfortunately, with such a low number of samples, typically 10 or fewer, it becomes impossible to investigate the root cause of wafer-level defects using conventional methods. This paper introduces a novel approach to finding the root cause of these rare defective chips for the first time in the semiconductor industry. Defective wafers are identified using rare customer-found chips and chip-level EDS (Electrical Die Sorting) data, and these newly identified defective wafers are termed vBADs (virtual bad wafers). The performance of root cause analysis is dramatically improved with vBADs. However, the chip-level analysis presented here demands substantial computing power. Therefore, MPP (Massive Parallel Processing) architecture is implemented and optimized to handle large volumes of chip-level data within a large architecture infrastructure that can manage big data. This allows for a chip-level defect analysis system that can recommend the relevant EDS test and identify the root cause in real time even with a single defective chip. The experimental results demonstrate that the proposed root cause search can reveal the hidden cause of a single defective chip by amplifying it with 90 vBADs, and system performance improves by a factor of 61.

Keywords: big data analytics; chip-level; customer-found defects; massive parallel processing; root cause detection

Citation: Kim, J.; Joe, I. Chip-Level Defect Analysis with Virtual Bad Wafers Based on Huge Big Data Handling for Semiconductor Production. *Electronics* **2024**, *13*, 2205. https://doi.org/10.3390/electronics13112205

Academic Editor: Francesco Giuseppe Della Corte

Received: 9 May 2024
Revised: 29 May 2024
Accepted: 4 June 2024
Published: 5 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In the semiconductor FAB (fabrication) process, wafers are produced through about 600 steps, and each chip is inspected through the EDS (Electrical Die Sorting) step to see if it satisfies the specification of electrical characteristics. The EDS step typically consists of around 3000 test items. Each test item individually tests each chip and yields a chip-level continuous value as the test result. Chips that do not meet the test quality requirements are classified as defective chips and are either screened out or undergo a repair process. As shown in Figure 1, the EDS step occurs after the FAB process, so the EDS test is conducted in the shape of a wafer, but each chip test is carried out individually. Chips that have passed this are packaged and shipped according to customer demand. After shipment, if a defect occurs, it is necessary to check whether the defective chip shows a significant difference in the EDS test item to prevent recurrence and to detect and control even if it recurs.

Figure 1 shows the overall semiconductor production procedure from FAB to customer. The following two activities are conducted with defective chips. The first is to find the

root cause of defects by finding the common FAB process of defective chips. The second is to analyze the correlation between defective chips and the EDS test. This correlation analysis allows for finding test conditions that can classify defects. We call it test coverage search, and through this, we estimate the defects–test correlation and adjust the test threshold value.

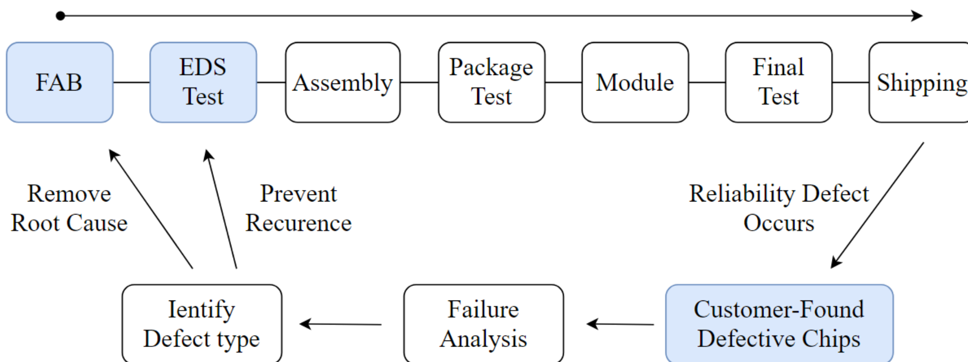

Figure 1. Overall procedure of semiconductor production. FAB (fabrication) and EDS (Electrical Die Sorting) are the most important steps in the semiconductor field. Customer-found defective chips are analyzed to find the root cause and prevent recurrence.

By definition, a reliability defect is a defect that causes the device to fail at $t > 0$, and a killer defect is a defect that causes the device to fail at $t = 0$. Price and Sutherland [1] use the term “latent defect” to refer to a reliability defect. Since the reliability defect goes undetected until the end of the line, it results in a possible loss of business and expensive failure analysis costs. Figure 2 shows that even the same type of defect can impact differently due to its size. Figure 3 shows that a reliability defect chip is not detected at the EDS step. We can detect latent defects using a neighboring chip’s data [2].

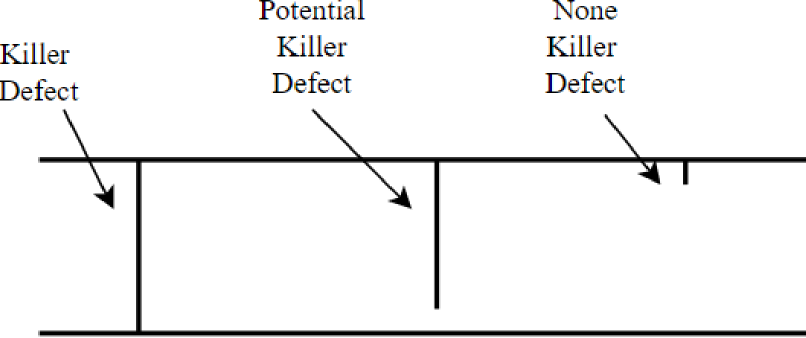

Figure 2. Potential latent reliability defect and killer defect. Potential latent defect (middle) is not detected at t = 0, but it will fail after being embedded in the customer’s electronic device.

Reliability defects occur in small quantities, and only extremely limited defects (1–10) are given after failure analysis steps such as Electrical Failure Analysis (EFA) and Physical Failure Analysis (PFA). That is a form of extremely imbalanced data, which makes it difficult to analyze. The problem of concern in this paper is quantifying the discrimination power of a continuous value for the pass/fail two-class to find test items showing significant differences in defective samples. Compared with the general two-class classification problem, the domain-dependent specificity of this problem is as follows.

1. Extremely imbalanced data;
2. Significant big data processing.

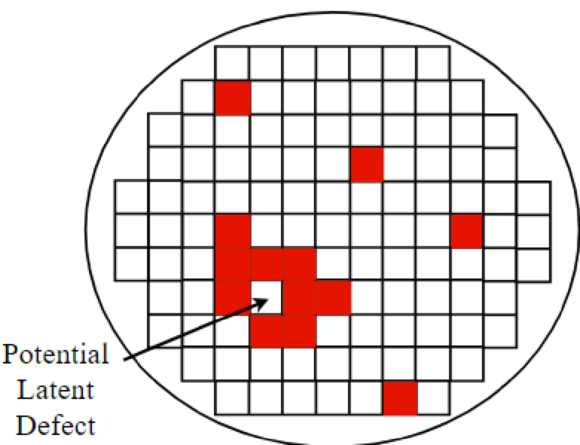

Figure 3. Potential latent reliability defect in EDS wafer map. Red chips are defective chips. The normal white chip surrounded by defective chips can be a reliability defective chip.

In the semiconductor field, effectively managing highly imbalanced data is crucial. Given the prevalence of extremely small defect occurrences, addressing this imbalance early on is essential for ensuring reliability. To tackle this challenge and facilitate real-time chip-level analysis, significant big data processing becomes indispensable in the semiconductor domain.

When analyzing imbalanced data in a semiconductor domain, we encounter two typical problems. First, if the root cause search in the FAB process is performed based on a small number of defect chips, too many pathways come to the surface, and it is hard to determine the root cause. Another problem is that when a correlation analysis with an EDS test is performed based on a small number of defects, the result value loses discrimination power due to the extremely imbalanced data. To mitigate such a problem, we chose Logistic Regression (LR) as a classifier, which can intuitively express defect correlation. We added the Synthetic Minority Over-Sampling Technique (SMOTE) as a preprocessing logic to improve the interpretation and intuition of R^2 [3]. Yi et al. [4] developed the MC-SMOTE oversampling technique for detecting wind turbine blade icing defects and improved predictive performance. The second problem is that the data size is increased by the number of net dies (1 wafer = 1900 chips), and a huge amount of time is required to obtain the result. To solve this problem, we adopted MPP (Massive Parallel Processing) architecture and converted the necessary data processing into a parallel processing form. The Massive Parallel Processing database adopts the SHARE-NOTHING distributed parallel processing flat architecture [5–7]. Timely analysis can be achieved by using MPP architecture to analyze rapidly generated semiconductor data. The rest of this paper is organized as follows. Existing methods for detecting semiconductor defects are reviewed in Section 2. The proposed methods to find the defect-relevant EDS test and the FAB root cause are reviewed in Section 3. The strategies for optimizing chip-level big data are reviewed in Section 4. The experimental result is summarized in Section 5. This paper concludes in Section 6.

2. Related Work

2.1. Semiconductor Defect

The proposed method is to select test items related to semiconductor defects. There are several similar studies that detect factors related to these semiconductor defects. Lee et al. [8] conducted a study to select critical process steps based on wafer-level measurement data. Missing values, random sampling, and imbalanced data that typically occur in the semiconductor domain were specified, and a data analysis methodology was used to overcome them. Lee and Kim [9] created a WBM (Wafer Bin Map) image using chip pass–fail binary data of EDS data and conducted a study on predicting a multi-label defect

based on WBM. Nam and Kim [10] studied the classification of low-yield wafers using the measure data of the FAB. The missing value problem was alleviated by utilizing the virtual metrology, and the imbalanced data problem was alleviated by using SMOTE preprocessing. Xu et al. [11] designed a data-driven adaptive wafer yield prediction VM model based on Gath–Geva fuzzy clustering (GGFC) and multitask learning deep belief network (GG-MLDBN). Those studies are wafer-level classification, which is an efficient method for detecting many defects. However, wafer-level analysis is inadequate to apply customer-found defective chips. This is because many cases of customer-found defects are random defects irrelevant to the wafer's data distribution and given an extremely small amount (1–10). In Section 3, the procedure of the test item investigation of the chip-level defect correlation is described.

2.2. Huge Data Handling

As semiconductor technology advances, cell size decreases, and interference increases. Accordingly, the need for more detailed chip-level investigation is growing. In addition, the amount of data is increasing over time. As the amount of data increases, the optimization of data processing time is the biggest requirement for recent semiconductor data analysis systems. There are similar studies on optimizing such large-scale data processing. Google implemented a MapReduce programming model that simplifies the parallel processing of large clusters [12]. Moreover, based on this MapReduce, the Hadoop Distributed File System (HDFS) is implemented and designed to run on commodity hardware [13,14]. Hadoop provides a distributed file system and a framework for analyzing very large data sets. Liu et al. [15] implemented motif discovery methods for large-scale time series healthcare data (MDLats) on MapReduce and Hadoop. MapReduce provides scalable and convenient ways to handle large-scale data. However, MapReduce and Hadoop do not integrate well with a relational database that stores large-scale data. Greenplum implemented the MPP database of a shared-nothing architecture [16–18]. This MPP database enables users to encapsulate special-purpose logic into declarative data processing as with MapReduce. Instead of using MapReduce, the productivity of developing applications is greatly improved by using SQL. Another big advantage of this implementation is that the existing Python R statistical library can be used in SQL as a database function. This useful functionality is called PL/R and PL/Python. Ma et al. [19] implemented a hybrid database solution including the MPP database, Hadoop, Storm, and a relational database to process huge data (TB) generated daily. In particular, the performance of Extract–Load–Transform of large-scale data was optimized based on the MPP database. The chip data in the semiconductor domain have the properties of lot ID, wafer ID, and chip ID; each process datum is joined with this ID. Processing large amounts of data inevitably requires a large amount of data join and degrades the entire application's performance. The MPP database stores data according to the distribution key, and when the distribution key and the join key match, the data join operation can be completely parallelized. With this characteristic, the MPP database strengthens in joining large amounts of data. For this reason, we adopted the MPP database as a data processing engine. In Section 4, strategies for optimizing massive data processing are described.

3. Proposed Methodology

In this section, we propose a methodology for finding EDS test items that are associated with a specific defect. Figure 4 shows the functional scheme of the proposed methodology. The proposed methodology starts with defective chips from the customer. Each defective chip has a unique ID, which we call chip coordinates. The chip coordinate is a unique key in the form of (r, w, x, y), where r is the Lot ID, w is the wafer ID, x is the horizontal position, and y is the vertical position, respectively. Based on these chip coordinates, the EDS data are extracted and applied to the preprocessing step, such as filtering and sampling. We have a vector of numerical value for each test item. These data are distributed, processed in parallel, and applied to the classifier model, and statistically quantified correlation

indicators such as R^2 and p-value are calculated. The quantified correlation index selects the test item related to the defect through technical review by an engineer with semiconductor domain knowledge. The correlation index can be used for low-yield wafer selection and test coverage validation. It can also be used to find latent defective wafers with a profile similar to the customer-found chip using the selected test item and the threshold value of a classification model. We call this newly found defective wafers the vBADs (virtual bad wafers).

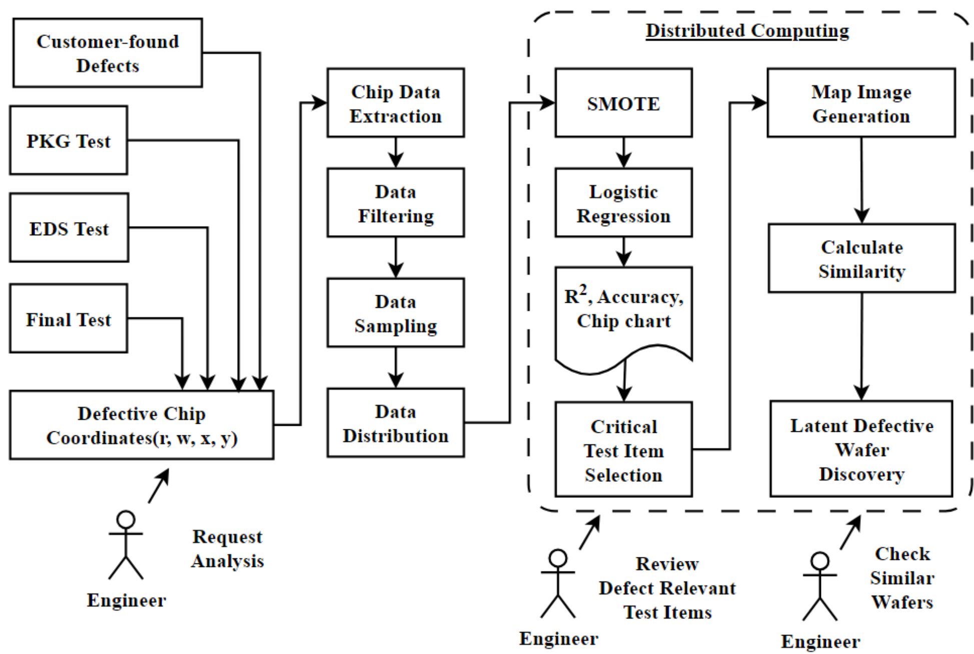

Figure 4. Functional scheme for selection of a critical EDS test item and latent defect discovery.

3.1. Data Extraction

This step extracts raw data based on the given defective chips. Since the defective chips are from products already delivered, test data must be stored for at least the past 3 years. However, the legacy system keeps only the data for the last 3 months due to the limitation of storage and expansion costs. Therefore, we built a separate distributed processing system and loaded it through the Hadoop ETL (Extract–Transform–Load) night batch process, and we are maintaining the chip-level test data of the last 3 years. It loads about 200 GB daily and uses a highly scalable storage and distributed processing database to improve the data load performance.

3.2. Data Filtering

Normal chips are also sampled and gathered for comparison with the customer-found chip data. We call them good chips. The selection of a good chip is important because it affects the accuracy of the classification model. We select good chips according to the following policy.

1. Chips with no history of defects in the EDS test process;
2. Chips belong to the same product as the defective chip.

The EDS test process consists of several steps. The good chip group includes only chips that pass through several test processes. This way, we can remove the outlier. We do not recommend the simple 3 Σ rule because the EDS test data do not follow the normal distribution. Semiconductor products are produced in several models according to the business needs, such as density, revision, etc.; each model has different test specifications. Therefore, it is necessary to select good chips that are in the same condition as the defective chips.

3.3. Data Sampling

The size of raw data ranges from 10 to 200 GB. In this step, the amount of processed data is reduced by applying random sampling to minimize the processing time. We empirically set the number of sampling chips as 20,000 to represent the data distribution while optimizing the data processing performance by reducing the data of good chips. In addition, two-phase sampling is performed to ensure the even distribution of chips belonging to a specific wafer. Wafer-based sampling (default: 200 wafers) was performed in the first phase, and chip-based sampling (default: 20,000 chips) was performed in the second phase. The sampling level of each wafer and chip is a system parameter that the user can change.

3.4. Over-Sampling

Reliability defects are defects that occur during customer use. These defective chips are given only in very small quantities, and this is intensified as EFA and PFA defect types are classified. Eventually, this forms an extremely imbalanced data set. Applying any existing classification model to extremely imbalanced data degrades the classification performance and discrimination power. To alleviate this limitation, we added the SMOTE as a preprocessing logic. By synthesizing customer-found chips, over-sampling is performed so that the ratio of defective chip count and good chip count approximates 1:1.

3.5. Classification

After the over-sampling, we have balanced N chips denoted by $\{(y_i, x_i), i = 1, \ldots, N\}$, where y_i is a pass–fail binary value, and x_i is the vector of test item continuous values, respectively. y_i is a binary variable, so y_i has a value of 0 or 1; y_i = 1 indicates that the i_{th} chip is a defective chip. x_i is a p-dimensional vector denoted by $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$, where p denotes the number of test items, and x_{ip} is the test item continuous value at the j_{th} test item of the i_{th} chip. In this step, the classification model is constructed by learning using the previously preprocessed pass–fail category balanced data. This constructed classification model calculates the correlation metric between the defect and each test item. LR was adopted as a classification model for the following reasons.

1. LR provides measures that quantify the correlation.
2. LR is easier to interpret and explain.
3. LR is resistant to overfitting in low-dimensional data.
4. LR works well with continuous and numerical variables.

Compared to other classification methodologies, LR provides general classification metrics and offers additional quantitative statistical data such as pseudo R^2, which indicates correlation based on features, p-values indicating statistical significance, and coefficients indicating the direction of correlation. These quantitative statistical data facilitate the explanation of correlations and serve as evidence for the subsequent handling of faulty data. Since the discovered correlations need to be explainable, LR necessitates a single definite and simple correlation factor rather than multivariate ones. In such low-dimensional data analysis, LR helps prevent difficult-to-explain overfitting. Additionally, LR assumes a linear relationship between the input variables and the output. Therefore, when the input variables are continuous and numerical, this assumption is likely to hold well.

To select a critical test item related to a defect from about 2000 test items in the EDS step, we need a metric to quantitatively indicate the correlation between each test item and defective chips. The LR model provides statistical metrics such as R^2 and p-value compared with other classification models and also provides useful statistical metrics of accuracy, sensitivity, and specificity through a validation process. In a linear regression model, the coefficient of determination, R^2, summarizes the proportion of variance in the dependent variable that is predictable from the independent variable. However, it is impossible to compute a single R^2 statistic in the LR model. Therefore, we adopted

McFadden's pseudo R^2, which is to compute approximated R^2 based on the likelihood ratio as following Equation (1).

$$McFadden's\ pseudo\ R^2 = 1 - \frac{ln\widehat{L}(M_{full})}{ln\widehat{L}(M_{intercept-only})} \quad (1)$$

The LR provides each feature's importance and finds an association's direction through an estimated metric. Compared to other models, the overfitting problem can be avoided by minimizing the number of variables. Multivariate classification using high-dimensional data can be used, but in reality, the results are overfitting and difficult to explain. Certainly, certain defects can be influenced by multiple test items. In such scenarios, there is an advantage in being able to delineate the defect scope and enhance classification accuracy precisely. However, substantiating the reasons for these influences necessitates physical analysis, consuming considerable time and semiconductor domain expertise. Moreover, as correlations can also be discerned for individual test items under the influence of multiple factors, this paper concentrates exclusively on analyzing the correlation of a single test item. For this reason, this paper assumes that defects are affected by only one test item. Based on the accuracy metric and R^2, we can obtain the Defect Correlation Index (DCI) by calculating the harmonic mean of accuracy and R^2 as the following Equation (2).

$$(Defect\ Correlation\ Index) = \frac{2 \cdot accuracy \cdot R^2}{accuracy + R^2} \quad (2)$$

Accuracy and R^2 are performance metrics of different methods, but they often yield similar values. However, when applied to semiconductor test items, it was observed that varying distributions of test item data and the values of input defective chips can lead to different outcomes. Compared to Decision Tree-based classifiers, Logistic Regression (LR) is more effective at working with continuous numerical data. LR's effectiveness with continuous numerical data stems from its ability to model linear relationships between input variables and the output. Since LR assumes a linear relationship, it is well suited for scenarios where the input variables are continuous and numerical. Because it can operate with significantly fewer defective data points, it helps minimize the risk of information loss from current test items. Moreover, since LR identifies high-risk wafers based on the distribution of continuous numerical data, it is better suited than Decision Trees for addressing the current challenge.

3.6. Critical Test Item Selection

In this step, the relevant top 10 test items are provided to the defect analysis engineer based on the statistical metric calculated in the previous step. The following is a policy that calculates the rank of defects in our methodology.

1. Exclude test items where 80% or fewer of all defective chips have values (e.g., excluding test items with only 7 out of 10 defective chips).
2. Exclude test items where defective chips are concentrated in the lower positions, particularly for smaller-is-better test items (e.g., fail bit count).
3. Rank the resulting test items based on the harmonic mean of their R^2 and accuracy values.

Statistical data regarding the R^2, p-value, estimate, accuracy, and sensitivity for each test item and visualization scatter chart and box plots were provided to help the defect analysis engineer's decision making. Based on the provided data, the defect analysis engineer uses domain knowledge to check whether the top 20 test items are actually related to the defect mechanism. Algorithm 1 shows the procedure that selects a critical test item based on provided defective chips C_1.

Algorithm 1 Critical Test Item Selection

Input: Defective chips C_1
Output: Top 10 Test item's Rank and Statistical Result
1: Filter data by product version based on C_1
2: Wafer and chip sampling, the sampled chip set C_2
3: Extract chip-level data based on C_2
4: Initialize test item data set T
5: Allocate test item's data into distributed cluster uniformly
6: **Do in Parallel:**
7: **for** `each test item in` T **do**
8: Initialize test item value-result set $S_1 = X_i, Y_i$
9: Apply SMOTE to get Balanced set $S_2 = X_i, Y_i$
10: Learn to construct LR model based on S_2
11: Calculate R^2 and accuracy using LR model
12: Calculate rank, DCI based on statistic index
13: **end for**
14: Sort by rank
15: **return** the Top 10 test item rank and statistical result

3.7. Latent Defect Wafer Discovery

In the previous step, we could quantitatively select the relevant test item. If the selected test item is verified through the domain knowledge of the defect analysis engineer, we can extend the defect analysis one step further. Based on the selected critical test item and its threshold value, the WBM (Wafer Bin Map) of each wafer can be generated, and by calculating the image similarity between the WBMs, wafers similar to defective wafers can be found. We call this vBADs (virtual bad wafers). This way, we can amplify similar behavioral wafers with a few chips. Such wafer amplification can help increase the root cause search hit rate in the FAB. Algorithm 2 shows the algorithm for the discovery of vBADs, in which we initialize the parameters using the statistical result of Algorithm 1. In Algorithm 1, the LR model yields the R^2, accuracy, cutoff value, and correlation coefficients. Using R^2 and accuracy, the critical test item was selected (T_s). We used the cutoff value as the pass/fail threshold (th_s) and the correlation coefficients as the failure direction (d_s), which means the direction of correlation. Then, we extracted the chip data corresponding to the T_s and grouped the wafer. By applying the previously initialized parameters th_s and d_s for each wafer, a binary image is created. The image of the wafer containing defective chips forms a bad wafer map (M_b), and the remaining wafer image forms a good wafer map group (M_g). By joining M_b and M_g without any condition, we create a Cartesian product of two groups (C_P) and calculate a similarity score based on each image pair in the C_P. Wafers that satisfy the similarity score threshold are added to the latent defect wafer list and returned. This way, the system provides the analysis engineer with a latent fail wafer list, wafer map image, and similarity score sorted in the order of similarity.

Algorithm 2 Latent Defective Wafer Discover

Input: Selected test item T_s, defective chips C
Output: Latent Fail Wafer List

1: Initialize defect threshold value th_s, fail direction d_s
2: Extract chip-level data of test item T_s
3: Grouping chip-level data by wafers W
4: Allocate wafer groups into distributed clusters uniformly
5: **Do in Parallel:**
6:     **for** `each test wafer in` W **do**
7:         Generate binary wafer map of defective wafer M_b
8:         Generate binary wafer map of normal wafer M_g
9:     **end for**
10: Generate Cartesian product of M_b and M_g C_P
11: **Do in Parallel:**
12:     **for** `each test pair in` C_p **do**
13:         Calculate the similarity score of each pair
14:     **end for**
15: Get virtual bad wafer by a similarity score
16: **return** Latent Defective Wafer list

4. Proposed System Architecture

We chose the MPP database to process large-scale semiconductor data within a limited time. The advantages of the MPP database include fast big data processing, MapReduce data processing through SQL, the ability to implement User-Defined Function (UDF) as a form of statistical programming scripts such as Python and R, and the reusability of its prebuilt data mining packages [16]. However, there are limitations, such as low latency and reduced parallel processing performance due to multiple-user access, so using it as a default data processing engine that stores general app data is difficult. As described above, there is an architectural driver that should improve the efficiency of large-scale data processing, concurrent user processing, and low latency. To alleviate this limitation, the user interface purpose web server is set, and the Distributed Task Queue is used as a data linker with the MPP database. Figure 5 is a high-level description of architecture and shows that the layers are divided as described above. The web server and relational database constitute the speed layer to handle concurrent user requests. A job that requires long-term data processing is delegated to the MPP database, which is delivered through a distributed task queue. The MPP database and task #N constitute the batch layer. The resulting data generated in the batch layer are transferred to the relational database of the speed layer to improve the response speed and concurrency data processing performance.

4.1. Distributed and Asynchronous Architecture

Data mining based on defective chips inevitably takes a long time. About 5 min of data processing time is required to analyze 50 GB of chip data. In the current implementation, the web server is used as the user interface, but the user has to wait in front of the web browser for 5 min, and the web server also has the main thread waiting, so other tasks such as user interface updates and user requests cannot be performed. To alleviate this limitation, we adopted Task Queues and Asynchronous Tasks. Using a Task Queue, load balancing and job scheduling are calculated based on subscribing workers on the task queue. Figure 6 shows the architectural view of the asynchronous execution.

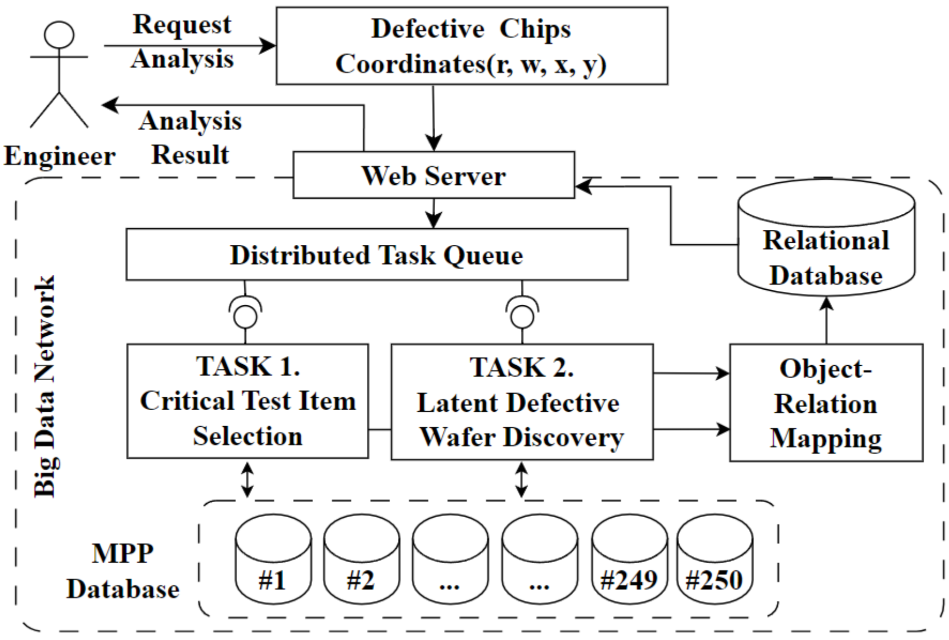

Figure 5. High-level system architecture.

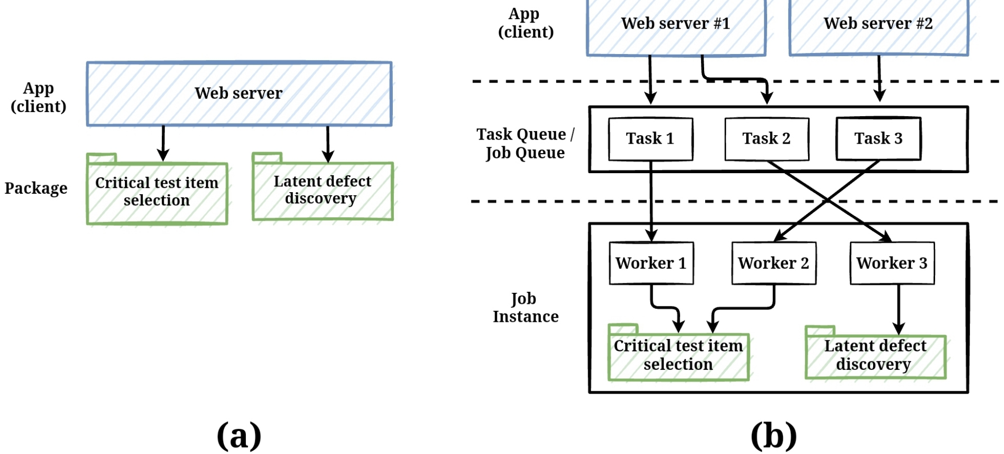

Figure 6. Distributed and asynchronous architecture. (**a**) Synchronous job execution. (**b**) Distributed and asynchronous job execution.

4.2. Data Multicast

If the maximum row count is much higher than the average, at least one segment has performed much more work than the others, and computational skew should be suspected for that operation. The MPP architecture was adopted as the backend data processing engine to obtain an optimized large-scale performance. In general, in the MPP architecture, data are divided and transferred to each worker node to process data, and the results of each worker node are aggregated in the master node. In this step, it is necessary to multicast the data equally to each worker node as a preparatory step for analyzing large amounts of data. There are two policies for distributing data: a distribution method based on a specific column value and a random distribution method. In our case, since the size of each test item is the same and execution performance is important, a random multicast method was chosen to distribute the data across all nodes uniformly.

5. Performance Evaluation

5.1. Experimental Design

This section illustrates the proposed method via a Metal Bridge defect case study. The number of defective chips is one. Nine days of chip-level data are extracted from the MPP database, whose data size is 66 GB. The EDS chip data contain 2200 test items, and each test item contains 16 million chips of continuous numeric data. After the wafer-level random sampling step, 200 wafers are selected randomly. After the chip-level random sampling, 20,000 chips are selected. The defective chip data are all utilized, while the utilization of good chips is aimed at comparing the distribution of good chip data with defective chip data. Through 200 wafer and 20,000 chip sampling, it is possible to obtain 100 chips per wafer, allowing for obtaining the distribution of good chip data that is not biased toward a specific wafer. In Section 5.2.2, it is noted that the analysis time increases proportionally with the amount of data, so it is being used as the default setting and can be customized for analysis according to the user's needs. We are using the MPP database to optimize the computation-intensive operation. Moreover, Hadoop is used to optimize data-loading to the MPP server. Table 1 summarizes the brief specifications of the two clusters used in this experiment.

Table 1. Experimental settings of MPP (Massive Parallel Processing) DB and Hadoop cluster.

Category	Parameter	Value
MPP DB	The number of physical servers	3 ea
MPP DB	The number of data nodes (segments)	250 ea
MPP DB	Clusters total SSD size	210 TB
MPP DB	Clusters total memory size	72 TB
MPP DB	The number of cluster's total CPU cores	512 core
Hadoop	The number of physical servers	16 ea
Hadoop	The number of data nodes	50 ea
Hadoop	Cluster's total SSD size	4.8 TB
Hadoop	Cluster's total memory size	4 TB
Hadoop	The number of cluster's total CPU cores	360 cores
Hadoop	External storage (NAS)	686 TB

To show performance improvement in root cause search, the Mean Reciprocal Rank (MRR) is used. MRR is a metric designed to measure the quality of ranked retrieval systems, which include search engines, recommendation systems, question-answering platforms, and more. MRR quantifies the position of the first relevant item in a ranked results list. It is particularly useful in scenarios where users seek a single relevant item within a larger set of results.

The MRR for a given set of queries is calculated as the average of the reciprocal ranks of the first relevant item for each query. If no relevant item is found for a query, its reciprocal rank is zero. MRR can be defined as follows, where $|Q|$ is the number of queries and $\mathrm{rank}(q)$ is the rank position of the first relevant result for a given query, respectively.

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\mathrm{rank}(q)} \quad (3)$$

5.2. Experimental Results

5.2.1. Case Study: Metal Bridge Defect

Table 2 is a correlation rank table obtained by investigating the metal bridge defect case. Rank 1 "Voltage X" test item's DCI is 0.99, R^2 is 0.91, and its accuracy is 0.99, which means statistically very relevant. Figure 7 shows additional visualizations of the first rank "Voltage X" test item. Using these visualizations, we can find abnormalities in the time-series trends of chip data or spatial patterns in the wafer map chart. We amplify defective wafer samples based on the selected critical item and threshold value found in the previous step. Figure 8 shows the box-plot diagram grouped by wafers. We classify

similar wafers to defective wafers as bad wafers and others as good wafers. Figure 9 shows the conceptual root cause analysis with amplified wafers.

Table 2. Correlation rank between Metal Bridge defect and EDS test item (top 5 of 2200 test items).

Rank	Test Step	Test Item	R^2	p-Value	Accuracy	DCI
1	Hot Test	Voltage X	0.99	0.001	0.99	0.99
2	Repair Test	Test A	0.54	0.683	0.91	0.68
3	Hot Test	Test B	0.53	0.004	0.94	0.68
4	Hot Test	Test C	0.47	0.769	0.90	0.61
5	Repair Test	Test D	0.47	0.010	0.87	0.61

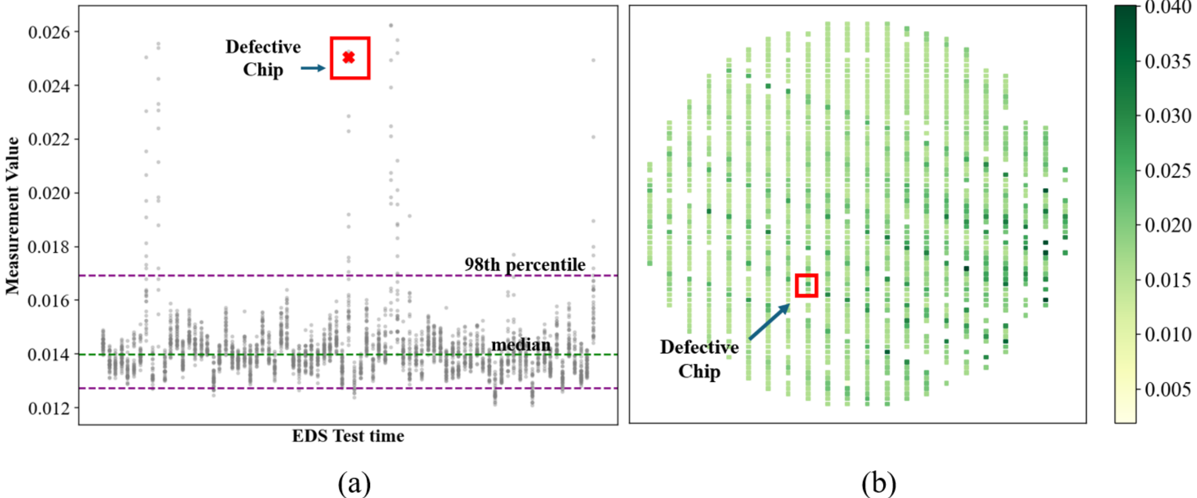

Figure 7. Visualizations for defective chip analysis. (**a**) The scatter chart of chips shows that a defective chip's measurement value is in the outlier range. (**b**) The wafer map chart shows that the defective chip is in the line patterns.

In this example, "C3" equipment in step "C" is the most suspicious root cause. For root cause analysis, we used the legacy in-house root cause search. The primary goal of our legacy in-house root cause analysis is to identify the common paths that bad wafers have taken. To pinpoint equipment that may be causing specific defects, we calculate the proportion of input bad wafers that pass through each equipment, which is known as the bad ratio. Similarly, we determine the proportion of good wafers that pass through each equipment, which is referred to as the good ratio. Finally, we calculate the difference between the bad wafer ratio and the good wafer ratio for each equipment, which we call the ratio gap. This metric helps us identify the root cause of the defects. The analysis engineer usually registers the defective wafers as "bad wafers" and the high yield as "good wafers" and executes the root cause analysis system. However, as we have discussed, since the number of defects in the real world is extremely limited, it is hard to find the root cause. Table 3 shows this limitation of a root cause search with a single defective wafer. Since there is only one bad wafer, all equipment has only two ratio gaps (76, 77%), and it is hard to determine the root cause. "E051" equipment at step "10" is a known root cause, but "E051" equipment ranked 404, and most of the equipment's ratio gap is identical or very close.

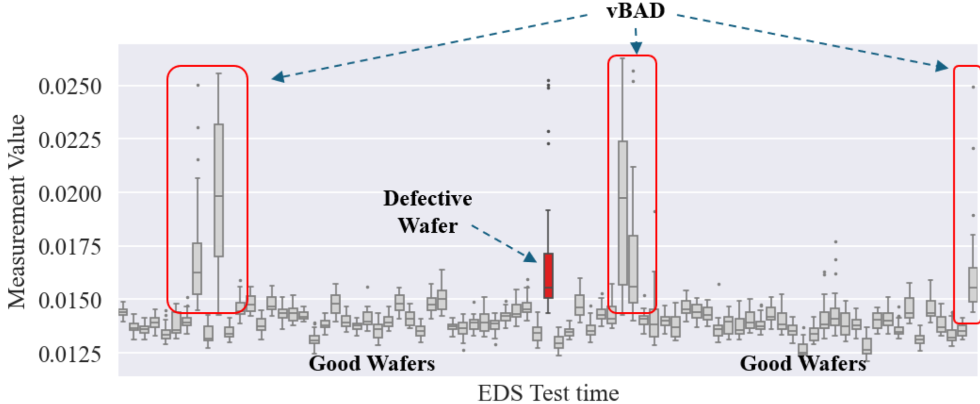

Figure 8. Similar defective wafers discovery. Using the selected test item and the defective chip measurement value, similar wafers can be found.

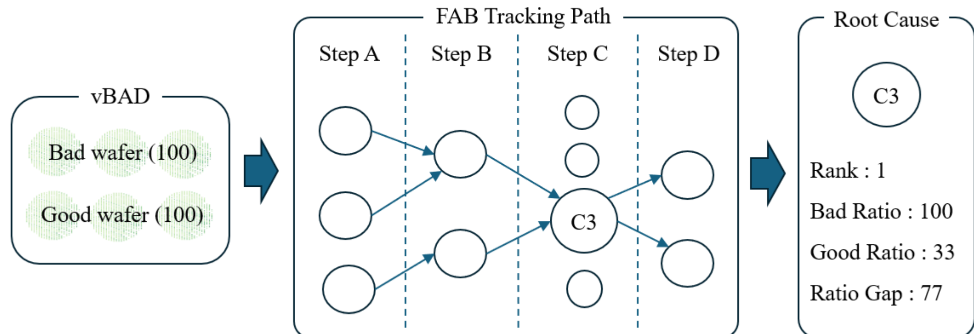

Figure 9. Root cause analysis with vBADs (virtual bad wafers).

Table 3. Root cause search with 1 defective wafer (rank: 404).

Rank	FAB Step	Equip.	Bad Ratio (%)	Good Ratio (%)	Ratio Gap (%)
402	70	E073	100 (1/1)	23 (23/100)	77.00
403	88	E198	100 (1/1)	23 (23/100)	77.00
404	10	E051	100 (1/1)	23 (23/100)	77.00
405	26	E032	100 (1/1)	24 (24/100)	76.00
406	92	E017	100 (1/1)	23 (23/100)	77.00

In contrast, Table 4 shows the result of the root cause search with 90 vBADs. "E051" equipment ranked first, with a bad ratio of 100%, a good ratio of 23%, and a gap of 77%. Figure 10 shows the overall root cause search performance by the number of vBADs. We utilized the Mean Reciprocal Rank (MRR) as a search performance metric. We fixed the number of good wafers to 100 to better observe the effect of the number of vBADs. The experimental result shows that the MRR of the root cause search with 90 vBADs reached 1.0, while the MRR of the root cause search with a single bad wafer is extremely low (0.0002). This proves that utilizing vBADs increases the accuracy of the root cause search. Additionally, the MRR of the root cause search with both 90 vBADs and 100 amplified good wafers is 1.0, while the MRR of the root cause search with 90 vBADs is 0.25. This proves that the root cause search with N-vBADs and amplified good wafers performs better than the root cause search only with N-vBADs. In summary, the experimental result shows that amplifying defective and good wafers can dramatically increase the accuracy of root cause search. The prior approaches solely detected low-yield wafers and applied them directly to the root cause search. However, when there is only a limited number of wafers or chips, the root cause search cannot provide the differentiated root cause. In the circumstance of a lack of defective chips, which is very common in the semiconductor industry, our proposed

method is essential and enables us to find the most correlated test and the most suspicious root cause, even with limited customer-found defects.

Table 4. Root cause search with 90 vBADs (rank: 1).

Rank	FAB Step	Equip.	Bad Ratio (%)	Good Ratio (%)	Ratio Gap (%)
1	10	E051	100 (90/90)	23 (23/100)	77.00
2	03	E489	98.88 (89/90)	32 (32/100)	66.88
3	05	E131	100 (90/90)	33 (33/100)	67.00
4	77	E123	100 (90/90)	52.52 (52/99)	47.48
5	36	E001	100 (90/90)	62 (62/100)	38.00

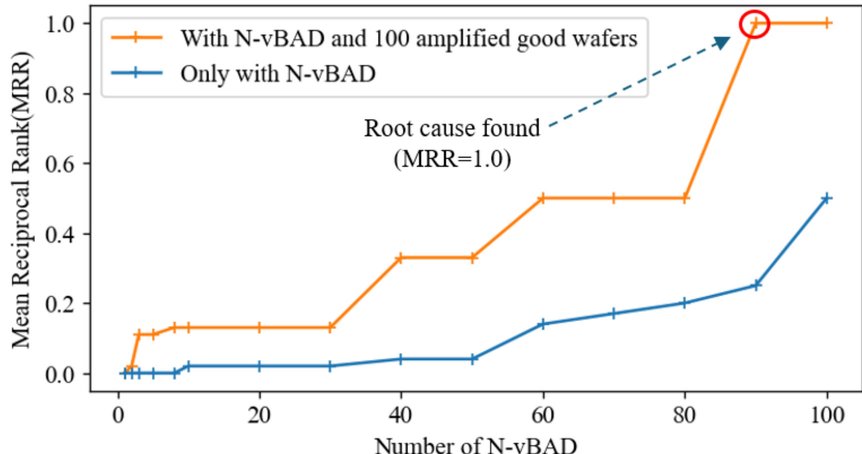

Figure 10. Root cause search performance increased by vBADs.

5.2.2. Time Complexity

Parameters that affect the execution time were confirmed through repeated experiments to ensure optimized performance. Figure 11 shows two factors that affect the performance of critical test item selection. The execution time increases proportionally to the raw data size and the number of test items. Therefore, the time complexity of the proposed algorithm empirically can be denoted as $O(nd)$, where n is the raw data size, and d is the number of test items. We guarantee a constant execution time by fixing these two parameters.

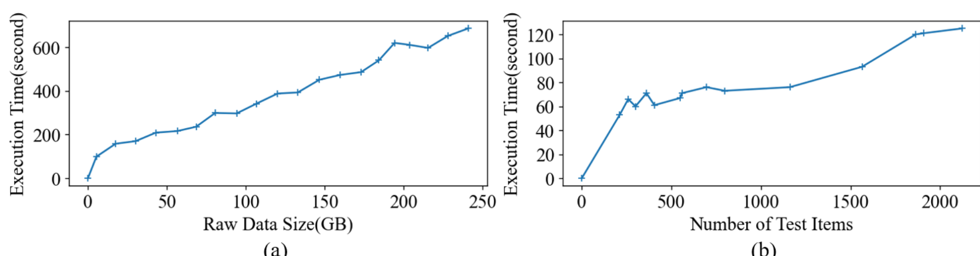

Figure 11. System parameters affecting the performance. (**a**) Data size affects the performance. 50 GB on average, 200 s elapsed. (**b**) The number of test items affects the performance. Test item counts are 2200, and 120 s elapsed.

5.2.3. System Performance Evaluation

Figure 12 illustrates how performance improves as the number of nodes in the MPP database increases. The MPP database provides a 61-fold performance boost in LR for selecting critical test items and a 54-fold improvement in joining large tables. The LR implementation is embedded as a function within SQL through an R script, allowing user-defined functions and external libraries in R or Python to be easily integrated and optimized

within the MPP database. This capability enables analysis engineers to conduct defect analysis and trace the root cause efficiently, even with just a few customer-found failures.

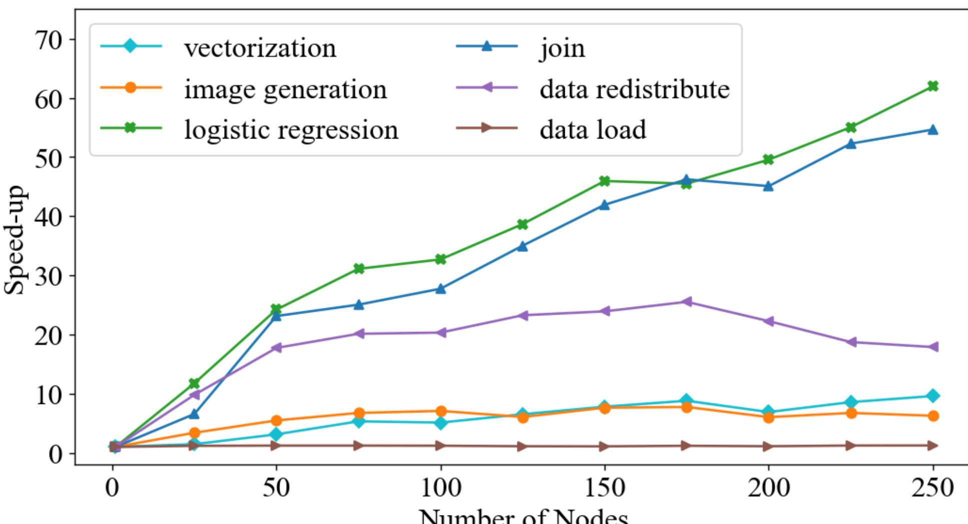

Figure 12. Performance improvement by MPP nodes counts.

6. Conclusions

This paper introduces a novel methodology to identify the most correlated EDS test and the most likely FAB root cause, even with a very limited number of chips, which is a first for the semiconductor industry. Previous methodologies had struggled to address customer-found defects due to their omission of chip-level data, which is crucial when dealing with rare defects. The complexity of chip data and the constraints of computational resources have made chip-level analysis time consuming or even unfeasible. However, chip-level analysis has gained importance in maintaining semiconductor quality as die sizes decrease especially when tackling customer-found defects. Our approach uses vBADs amplification to enhance the root cause search performance despite the limited number of defective chips. We employed an MPP database to manage large volumes of chip-level data efficiently. Experiments using real-world EDS data and customer-found defects involved amplifying bad and good wafers, up to 100 each, and applying them to root cause searches to pinpoint the most likely root cause. The MRR metric was used to assess the performance of our root cause search methodology, showing a 5000-fold improvement with 90 amplified vBADs compared to the original customer-found defect data. Additionally, we assessed the system's performance improvement across different types of data operations. Key statistical and join operations showed up to a 61-fold increase in performance using an MPP database compared to a single-node database.

Author Contributions: Conceptualization, J.K.; Methodology, J.K. and I.J.; Software, J.K.; Validation, J.K.; Formal Analysis, J.K.; Investigation, J.K.; Resources, I.J.; Data Curation, J.K. and I.J.; Writing—Original Draft Preparation, J.K.; Writing—Review and Editing, J.K. and I.J.; Visualization, J.K.; Supervision, I.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data that support the findings of this study are available on request from the corresponding author, I.J. The data are not publicly available because they include information that could compromise the privacy of the study participants.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

FAB	fabrication
EDS	Electrical Die Sorting
vBADs	virtual bad wafers
MPP	Massive Parallel Processing
EFA	Electrical Failure Analysis
PFA	Physical Failure Analysis
LR	Logistic Regression
SMOTE	Synthetic Minority Over-Sampling Technique
WBM	Wafer Bin Map
ETL	Extract–Transform–Load
DCI	Defect Correlation Index
UDF	User-Defined Function
MRR	mean reciprocal rank

References

1. Price, D.W.; Sutherland, D.G. Process Watch: The Most Expensive Defect; Part 2. Solid State Technology (On-Line and Print Editions). 2015. Available online: https://sst.semiconductor-digest.com/2015/07/process-watch-the-most-expensive-defect-part-2/ (accessed on 1 June 2020).
2. Robinson, J.C.; Sherman, K.; Price, D.W.; Rathert, J. Inline Part Average Testing (I-PAT) for automotive die reliability. In Proceedings of the Metrology, Inspection, and Process Control for Microlithography XXXIV, San Jose, CA, USA, 24–27 February 2020; Adan, O., Robinson, J.C., Eds.; International Society for Optics and Photonics, SPIE: Bellingham, WA, USA, 2020; Volume 11325, pp. 50–59.
3. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
4. Yi, H.; Jiang, Q.; Yan, X.; Wang, B. Imbalanced Classification Based on Minority Clustering SMOTE with Wind Turbine Fault Detection Application. *IEEE Trans. Ind. Inform.* **2020**, *17*, 5867–5875. [CrossRef]
5. Patel, A.; Birla, M.; Nair, U. Addressing big data problem using Hadoop and Map Reduce. In Proceedings of the 2012 Nirma University International Conference on Engineering (NUiCONE), Ahmedabad, India, 6–8 December 2012; pp. 1–5.
6. Tsakalozos, K.; Verroios, V.; Roussopoulos, M.; Delis, A. Time-Constrained Live VM Migration in Share-Nothing IaaS-Clouds. In Proceedings of the 2014 IEEE 7th International Conference on Cloud Computing, Anchorage, AK, USA, 27 June–2 July 2014; pp. 56–63.
7. Wang, X.; Yang, L.; Xie, X.; Jin, J.; Deen, M. A Cloud-Edge Computing Framework for Cyber-Physical-Social Services. *IEEE Commun. Mag.* **2017**, *55*, 80–85. [CrossRef]
8. Lee, D.H.; Yang, J.K.; Lee, C.H.; Kim, K.J. A data-driven approach to selection of critical process steps in the semiconductor manufacturing process considering missing and imbalanced data. *J. Manuf. Syst.* **2019**, *52*, 146–156. [CrossRef]
9. Lee, H.; Kim, H. Semi-Supervised Multi-Label Learning for Classification of Wafer Bin Maps with Mixed-Type Defect Patterns. *IEEE Trans. Semicond. Manuf.* **2020**, *33*, 653–662. [CrossRef]
10. Nam, W.S.; Kim, S.B. A Prediction of Wafer Yield Using Product Fabrication Virtual Metrology Process Parameters in Semiconductor Manufacturing. *J. Korean Inst. Ind. Eng.* **2015**, *41*, 572–578. [CrossRef]
11. Xu, H.W.; Qin, W.; Lv, Y.L.; Zhang, J. Data-Driven Adaptive Virtual Metrology for Yield Prediction in Multibatch Wafers. *IEEE Trans. Ind. Inform.* **2022**, *18*, 9008–9016. [CrossRef]
12. Dean, J.; Ghemawat, S. MapReduce: Simplified Data Processing on Large Clusters. In Proceedings of the OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, USA, 6–8 December 2004; pp. 137–150.
13. Shvachko, K.; Kuang, H.; Radia, S.; Chansler, R. The Hadoop Distributed File System. In Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Lake Tahoe, NV, USA, 6–7 May 2010; pp. 1–10.
14. Borthakur, D. The Hadoop Distributed File System: Architecture and Design. *Hadoop Proj. Website* **2007**, *11*, 21.
15. Liu, B.; Li, J.; Chen, C.; Tan, W.; Chen, Q.; Zhou, M. Efficient motif discovery for large-scale time series in healthcare. *IEEE Trans. Ind. Inform.* **2015**, *11*, 583–590. [CrossRef]
16. Waas, F. Beyond Conventional Data Warehousing—Massively Parallel Data Processing with Greenplum Database—(Invited Talk). In Proceedings of the BIRTE, Auckland, New Zealand, 24 August 2008.
17. Lyu, Z.; Zhang, H.H.; Xiong, G.; Guo, G.; Wang, H.; Chen, J.; Praveen, A.; Yang, Y.; Gao, X.; Wang, A.; et al. Greenplum: A Hybrid Database for Transactional and Analytical Workloads. In Proceedings of the 2021 International Conference on Management of Data, Virtual Event, 20–25 June 2021; pp. 2530–2542.

18. Gollapudi, S. *Getting Started with Greenplum for Big Data Analytics*; Packt Publishing Ltd.: Birmingham, UK, 2013.
19. Ma, S.; Xiao, H.; Xu, B.; Tao, R.; Xie, F.; Zeng, D.; Wang, T. Bank Big Data Architecture Based on Massive Parallel Processing Database. In Proceedings of the 2018 15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN), Yichang, China, 16–18 October 2018; pp. 93–99.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.